In compliance with the
Canadian Privacy Legislation
some supporting forms
may have been removed from
this dissertation.

While these forms may be included
in the document page count,
their removal does not represent
any loss of content from the dissertation.

UNIVERSITY OF ALBERTA

KNOWLEDGE EXTRACTION ON SOFTWARE ENGINEERING DATA

by

EFE IGBIDE

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of Master of Science

DEPARTMENT OF ELECTRICAL & COMPUTER ENGINEERING

Edmonton, Alberta

Fall 2003

National Library of Canada
Bibliothèque nationale du Canada

Acquisitions and Bibliographic Services
Acquisisitons et services bibliographiques

395 Wellington Street
Ottawa ON K1A 0N4
Canada

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence
ISBN: 0-612-87788-4
Our file Notre référence
ISBN: 0-612-87788-4

The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

L'auteur conserve la propriété du droit d'auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou aturement reproduits sans son autorisation.

Canada

University of Alberta

Library Release Form

Name of Author:	Efe Igbide
Title of Thesis:	Knowledge Extraction on Software Engineering Data
Degree:	Master of Science
Year this Degree Granted:	2003

Permission is hereby granted to the University of Alberta Library to reproduce single copies of this thesis and to lend or sell such copies for private, scholarly or scientific research purposes only.

The author reserves all other publication and other rights in association with the copyright in the thesis, and except as herein before provided, neither the thesis nor any substantial portion thereof may be printed or otherwise reproduced in any material form whatever without the author's prior written permission.

e

8

Date: MAY 16, 2003

University of Alberta

Faculty of Graduate Studies and Research

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research for acceptance, a thesis entitled Knowledge Extraction from Software Engineering Data submitted by Efe Igbide in partial fulfillment of the requirements for the degree of Master of Science.

Supervisor: Dr. Marek Reformat

Dr. Witold Pedrycz (Internal)

Dr. Eleni Stroulia (External)

Date: May 12, 2003

Abstract

Software maintenance is one of the most time and effort-consuming phases of the software development life cycle. Maintenance engineers need tools and methods to support scheduling and perform software maintenance tasks. In this study, a comprehensive multi-model estimation system based on evidence theory and several rule-based models is proposed. The use of evidence theory enables the determination of confidence levels for the generated rules and estimation results obtained from the system. Applicability of the proposed system is validated by its usage to knowledge extraction and prediction for the well-known data sets.

This study comprises three detailed cases. The first deals with the estimation of the number of software components, which have to be examined during elimination of defects. The second case is concerned with predicting the time needed in isolation of defects, and the third with the effort needed for defect removal.

Acknowledgements

In carrying out my studies here at the University of Alberta, a number of people and organizations have assisted me, been there for me and most importantly, encouraged me in my study. With a few lines, I would like to pay a tribute to them:

Special thanks to my supervisor Dr. Marek Reformat for his guidance and encouragement over the course of this work, as well as his painstaking efforts to provide timely and detailed markups to the thesis draft and providing numerous key ideas that were used throughout this research. Indeed, I owe the successful completion of this work entirely to him.

My fiancée Maimouna, my parents Josiah and Juliana, my brothers and sisters Vivienne, Philip, Paul, Ajiri, Igho and Kome, and my nephew Christopher, nieces Naomi and Oke for their love and support.

The invaluable sources of funds that I had available to me: Natural Sciences and Engineering Research Council (NSERC) of Canada and Alberta Software Engineering Research Consortium (ASERC) Edmonton.

I would like to thank the Staff and students (especially members of the Quantitative and Soft-Computing based Software Engineering (QSSE) Group Edmonton) for their tremendous support and for providing a conducive environment in which to carry out my studies.

Finally, much thanks to my bosom friends Ayo, Ubaka, Segun, Istihad and Niyi for being there for me on this journey.

Table of contents

1. INTRODUCTION 1

1.1 Introduction 1
1.2 Thesis Objectives 2
1.3 Thesis Contributions 3
1.4 Research Methodology 3
1.5 Thesis Outline 4

PART 1: CONCEPT 6

2. THEORETICAL BACKGROUND 7

2.1 Software Maintenance 7
2.2 Knowledge Models for prediction and estimation 9
2.2.1 Modeling Concept 10
2.2.2 Rule-based Models 11
2.2.3 Multiple Models 13
2.3 Evidence Theory 15
2.3.1 Basic Concepts 15
2.3.2 The Transferable Belief Model 16
2.3.3 Basic Belief Assignments 16

3. MULTI-MODEL ESTIMATION SYSTEM 21

3.1 Concept 21
3.2 Model Generation Methods 23
3.3 Generation and Validation of Rule Based Models 23
3.4 Inference Engine 26

PART II: KNOWLEDGE EXTRACTION 31

4 UCI MACHINE LEARNING REPOSITORY DATA SET 32

4.1 Heart Disease Data (Cleveland Clinic Foundation) 32
4.2 Lymphography Data 33
4.3 Breast Cancer Data 34

5 APPLICATION OF MULTI-MODEL ESTIMATION SYSTEM TO ML DATA 35

5.1 Breast Cancer Data 35
5.1.1 Model Building 35
5.1.2 Validation of Rules 37
5.1.3 Analysis of IF-THEN Rules 42
5.1.4 Classification of Breast Cancer Cases 43
5.2 Lymphography Data 45
5.2.1 Model Building 45

5.2.2 Validation of Rules 47
5.2.3 Analysis of IF-THEN Rules 49
5.2.4 Prediction of Lymph Nodes Problems 50
5.3 Heart Disease Data 51
5.3.1 Model Building 51
5.3.2 Validation of Rules 54
5.3.3 Analysis of IF-THEN Rules 58
5.3.4 Prediction of Heart Disease 59

PART III: SOFTWARE MAINTENANCE 61

6 SOFTWARE MAINTENANCE DATA SET 62

6.1 Data Selection and Extraction 62
6.2 Data Pre-processing 63

7 ISOLATION OF SOFTWARE DEFECTS 66

7.1 Model Building 66
7.2 Validation of Rules 68
7.3 Analysis of IF-THEN Rules 71
7.4 Prediction of Time for Isolation of Defects 72

8 NUMBER OF COMPONENTS EXAMINED 74

8.1 Model Building 74
8.2 Validation of Rules 78
8.3 Analysis of IF-THEN Rules 82
8.4 Prediction of Number of Components to be Examined 84

9 EFFORT FOR ELIMINATION OF SOFTWARE DEFECTS 87

9.1 Model Building 87
9.2 Validation of Rules 91
9.3 Analysis of IF-THEN Rules 95
9.4 Prediction of Efforts for Elimination of Defects 97

10 CONCLUSIONS AND FUTURE WORK 99

List of Tables

Table 1. Rules with their bbm 27
Table 2. Basic belief masses assigned to each possible category by satisfied rules 29
Table 3. Basic belief masses obtained by the conjunctive combination 29
Table 4.Pignistic probability (BetP) value 29
Table 5.Heart Disease Attributes, Values and Information 32
Table 6.Lymphography Data Attributes and Values 33
Table 7. Breast Cancer Attributes and Values 34
Table 8.Original and updated values of bbm for rules generated by C5 38
Table 9.Original and updated values of bbm for rules generated by 4C 39
Table 10.Original and updated values of bbm for rules generated by GB 40
Table 11.Rules satisfied by the five "confusing" data points 44
Table 12.BetP values for the seven "confusing" data points 44
Table 13.Original and updated values of bbm for rules generated by C5 47
Table 14.Original and updated values of bbm for rules generated by 4C 48
Table 15.Original and updated values of bbm for rules generated by GB 48
Table 16.Rules satisfied by the three "confusing" data points 51
Table 17.BetP values for the three data points 51
Table 18.Original and updated values of bbm for rules generated by C5 55
Table 19.Original and updated values of bbm for rules generated by 4C 56
Table 20.Original and updated values of bbm for rules generated by GB 57
Table 21.Rules satisfied by the three "confusing" data points 60
Table 22.BetP values for the three "confusing" data points 60
Table 23.Data attributes to estimate Time needed for Isolation of Defect 63
Table 24.Data attributes to estimate Number of Components Examined 64
Table 25.Data attributes to estimate Effort needed for Elimination of Defect 65
Table 26.Original and updated values of bbm for rules generated by C5 69
Table 27.Original and updated values of bbm for rules generated by 4C 69
Table 28.Original and updated values of bbm for rules generated by GB 70
Table 29.Rules satisfied by the eight "confusing" data points 73
Table 30.BetP values for the eight "confusing" data points 73
Table 31.Original and updated values of bbm for rules generated by C5 78
Table 32.Original and updated values of bbm for rules generated by 4C 79
Table 33.Original and updated values of bbm for rules generated by GB 81
Table 34.Rules satisfied by the seven "confusing" data points 85
Table 35.BetP values for the seven "confusing" data points 85
Table 36.Original and updated values of bbm for rules generated by C5 92

Table 37.Original and updated values of bbm for rules generated by 4C 92
Table 38.Original and updated values of bbm for rules generated by GB 94
Table 39.Rules satisfied by the twelve "confusing" data points 97
Table 40.BetP values for the first six data points 98
Table 41.BetP values for the last six data points 98

List of Figures

Figure 1.Structure of a Rule-Based Model..... 11
Figure 2.Multi-model Estimation System 24
Figure 3. Structure of Inference Engine 28

1. Introduction

1.1 Introduction

One of the many difficult aspects of software engineering is the difficulty of extracting knowledge about the system studied from collected data. Software organizations have often collected volumes of data in hope of better understanding their processes and products. Useful information has been extracted from those large volumes of data, but it is commonly believed that large amounts of useful information remain hidden in software engineering databases [16].

Many data mining algorithms have been proposed for finding information hidden in software data. Data mining is a technique used for discovering patterns, relationships, and hidden information in data and has proved to be one of the tools of choice for exploring software engineering data. One of the difficulties in using data mining is a lack of knowledge of how to combine several attributes to get more accurate mining results. Growing awareness that software engineering needs to improve its products and incorporate more rigors into its processes has recently led to increased interest in knowledge extraction.

The challenge of extracting knowledge from data draws upon research in statistics, databases, pattern recognition, machine learning, data visualization, optimization, and high-performance computing, to deliver advanced business intelligence and web discovery solutions.

Data mining is an umbrella term that applies to a multitude of techniques for extracting from massive quantities of information various types of important, interesting, or unexpected phenomena. Because the information of interest is of a wide variety of natures, and because the type of phenomena which we seek varies and often is ill defined, many diverse technologies must be developed and applied in novel ways.

The applications that motivate our work are characterized by problems in which of a huge quantity of information presented, only a small fraction is of particular interest. Since it is often costly to determine whether an individual item is truly interesting (e.g., by investigating its source), resource

limitations require that we prioritize our effort so that only the items of interest are pursued. By providing a prioritization rather than simply a classification, we can pursue the highest ranked items until time, money, or interest is exhausted, and the remaining items implicitly are discarded.

We are engaged in an ongoing effort to develop prioritization systems with emphases on the algorithmic and statistical components of the solution. The value of such prioritization systems is ever increasing in the context of today's information explosion, where one is often flooded with an overwhelming quantity of information and must make snap decisions under uncertainty. Important application areas include computer security, telecommunications, document retrieval, radio astronomy, and biomedical testing.

1.2 Thesis Objectives

The goal of this research is to use data mining for prediction purposes in software engineering. This study seeks to develop models that will be effective and accurate in prediction with the highest possible confidence level. The study addresses both the issues of prediction and confidence levels. It tries to seek a trade off between achieving a high prediction level and having a high confidence that the prediction is good.

This research is set out to take the advantage of the synergy effect by combining knowledge extracted from data using different techniques. The study also introduces the concept of knowledge reuse with beliefs and uncertainty. The extent of knowledge reuse can facilitate allocation of knowledge base, guide the choice of knowledge bases and facilitate knowledge management system design while improving diagnostic problem solving. We take different predictions into account and choose the best prediction suitable from the point of view of all data points rather than a single one.

This study also looks at some maintenance issues relating to the application of multi-models in the prediction of the effort and time it takes to eliminate software defects and the number of components examined in the prediction process. In this study, knowledge that tells us more about

the relation between the attributes and the predicted class and to what extent each of them affects the confidence level and the predicted class being extracted.

1.3 Thesis Contributions

In this study, a range of different tools is used to develop a system based on Evidence theory [25] for prediction and also to indicate the confidence level of the prediction. The system takes advantage of knowledge extracted from the various tools. In this thesis, the objective is not to look only at the knowledge extraction from the prediction aspect but also at the confidence levels, which can be assigned to obtain predictions.

UCI Machine Learning data was used to prove the usefulness of the model. It was also used to gain confidence that the model performs to specification. After gaining confidence using the UCI Machine Learning data, the proposed model was applied to Software Maintenance data to predict the effort required to eliminate software defects, time required for the isolation of software defects and the number of components to be examined. From this study, some important rules were extracted in these three areas that can be used for prediction purposes with high confidence for the data set used.

Also, the classification rate has been increased by the used of this multi-model estimation approach. It should also be noted that this approach has also help in resolving conflicts in classification from these three models namely 4CruleBuilder, well-known See5 and the Genetic Based model. For instance, in cases where the prediction on a particular data point is different for each model, the multi-model system helps indicate which group it actually belongs to with a given probability. Estimation systems, extracted knowledge and results of prediction processes have been described and investigated.

1.4 Research Methodology

The approach proposed in the thesis addresses the issue of the use of a variety of **IF-THEN** rules and methods of their development, as well as the lack of indicators of confidence levels of results obtained from **IF-THEN** based models. Once confidence levels for the various rules are

established, evidence based method is used to combine them based on their confidence levels. The main idea of the approach is to use a number of rule-based models and belief functions from evidence theory to assert and validate the confidence level of rules. Transferable Belief Models [27] built on evidence theory is then used to reason about the membership of a data point to a specific class and the confidence the result.

UCI Machine Learning data is used to prove the concept. The validation of the concept is performed by using software maintenance data to predict the effort required to eliminate software defects, the time required for the isolation of those defects and the number of components that needs to be examined.

1.5 Thesis Outline

Chapter 2 presents a detailed review of software maintenance, knowledge models for prediction and estimation, and evidence theory. It also presents a brief overview of previous research efforts in the area of software maintenance, multiple models and evidence theory and describes the modeling concept used in this thesis and the basic concepts of evidence theory.

Chapter 3 describes the process of developing the multi model estimation system and the various ways the models can be generated and combined. It also covers the generation and validation of the rule based models and how they relate to the inference engine. It covers the outline of the approach used, the main components of the system comprising of a sets of **IF-THEN** rules and the use of Belief Functions.

Chapter 4 describes the UCI machine learning repository data set used to prove the usefulness of the approach. The three sets of data described here are the heart disease data set, lymphography data set and the breast cancer data set.

Chapter 5 gives a description of the application of multi-model estimation system to the UCI machine learning data set.

Chapter 6 describes the software maintenance data set used in this thesis. This chapter illustrates the data selection and extraction process. It also describes how the data is preprocessed so the models can use it.

Chapters 7, 8 and 9 describe isolation of software defects, no of components examined and the effort needed for elimination of software defects. It also covers the model building, validation of rules, analysis of **IF-THEN** rules and also prediction of the number of components examined.

Chapter 10 comprises the conclusion and contributions of this research study, as well as limitations and future work.

PART 1: CONCEPT

2. Theoretical Background

2.1 Software Maintenance

Software maintenance is an important concern for organizations but conventional software engineering tends to focus only on the design and implementation of a product, even though that forms a small part of the software life cycle. On a life-cycle basis, more than two-thirds of software costs occur after a system has been implemented [4,17,26]. It is thought that a large portion of these maintenance costs results from poor practices in software design and development [23]. Because the lifetime of software systems is often measured in decades [35], the long-term costs of supporting poor quality software can be substantial.

Most of the time, the decision whether to change or enhance a software system is determined by the costs and benefits of that particular maintenance task. However, there are situations where maintainers of an existing system are left with no choice but to change the system at any cost. Such situations may arise as a result of changes to the operating system in use. Thus, it is essential to know the time it takes to fix a particular problem as this can help in estimating and predicting the cost of maintenance. Predictions of effort needed to carry out software maintenance tasks may be an important input to maintenance managers when planning maintenance activities and performing cost/benefit analysis [13].

Many organizations want to prepare reliable schedules of maintenance tasks. Such schedules can lead to on-time realization of the tasks and better management of resources. It is an important issue, especially in the case when software maintenance tasks account for more than half of the typical software budget [12,31]. In response to that, the software industry is exhibiting an increased interest in the benefits yielded by improved software maintenance processes. To support maintenance activities and make them more efficient software engineers use a number of different tools. These are tools for model-based software component analysis, metrics application, measurement result presentation, and statistical analysis and evaluation. Software maintainers need to understand relationships between attributes of software components and maintenance efforts required by a software system built with these components. Gained knowledge would increase

understanding of the impact of software component attributes, such as size of code, complexity, functionality, etc., on efforts associated with maintenance activities.

There are four different categories of software maintenance: corrective – it involves changing software to remove defects; adaptive – it leads to changing software due to changes in software operating environment; perfective – it embraces activates which lead to improvement of maintainability, performance, or other attributes of software; and preventive – it is defined as maintenance performed for the purpose of preventing problems before they happen. The corrective software maintenance is associated with activities related to elimination of software defects. This process is a key factor in ensuring timely releases of software updates and high quality of software. Software maintainers need tools and systems, which support these activities. Besides the tools that are directly related to correction of defects, there is also a need for systems that support decision-making tasks leading to preparation of schedules and plans of maintenance processes. Such systems should not only provide a quantitative estimation but also give indication about plausibility of the estimation, and supply users with knowledge leading to justification of the provided estimation. Therefore, it is desirable to have a tool that retrieves knowledge about relationships between attributes describing software, and factors that directly or indirectly influence defect elimination activates. This would help software maintainer to better understand mechanisms that affect time and resourses needed for these actions. There are a number of important questions related to removal of defects from software systems:

- Does defect elimination depend on size of software components?
- Does it depend on a functionality of software components?
- Does it depend on a phase where defects have been introduced?
- What are the factors that influence time needed for correction of a single defect?
- What kind of relations can be found from software maintenance data?
- How confident someone can be about found dependencies?

2.2 Knowledge Models for prediction and estimation

Like any human-intensive production/engineering activity, software development needs reliable techniques to plan resource expenditures and monitor, assess, and control product quality [6].

The foregoing richly demonstrates the necessity to predict project expenditures and monitor major deviations through the construction of heuristics and accurate prediction models. Reliance on these models will help detect deviations and implement useful remedies. A number of machine learning techniques have proven helpful as prediction models. Such techniques include decision trees, classification rules, extending linear classification: support vector machines, instance-based learning, numeric prediction, clustering.

Knowledge models are used to predict the outcomes of various control actions thus providing a basis for selecting the best control action. The model-based technique is a very powerful knowledge representation. This concept can also be applied to other domains. Models can be constructed to capture the gist of software processes. These models can then be manipulated to predict the effects of various actions. Built as part of a knowledge base system, the models can predict outcomes based on different data types and combinations scenarios. This type of reasoning is very useful as part of a sophisticated decision support system (DSS)[18]. One of the key challenges is to ensure that the model captures the most important characteristics of the data being analyzed.

Model prediction and estimation is an iterative process. The first model may not always be optimal. It may require changes in the data or the parameters of the model. There are several techniques that professionals can employ in their data mining models and each technique has its advantages and disadvantages.

All of the methods used will provide answer but the quality of the results will have a direct relationship with the technique used and how good is the available data. Therefore there is need for proper data representation. It is important to understand critical that one understands the available techniques so they can align the techniques with goals.

2.2.1 Modeling Concept

All prediction models aim at predicting outcome on the basis of a given set of variables: they estimate what should be the outcome of a given situation, with a certain condition defined by the values of the given set of variables. This sentence is a very simple explanation what a prediction model is. The steps that have to be preformed during development of such a model are:

- selection of the outcome attribute;
- selection of predictor variables;
- data collection;
- assembly of the model;
- validation of the model;
- model updates and modifications.

A generic nature of the steps means that prediction models can be built and applied in many areas of engineering.

In the case of software engineering, prediction models that support maintenance activities are used for many years. Main areas of investigations are related to:

- predicting a number of defects in the system;
- estimating reliability of the system in terms of time to failure;
- understanding the impact of design and testing processes on a number of defects and frequency of failures.

A wide range of prediction models has been proposed. The most popular ones are reliability models. For example, models for predicting a number of defects [10] or fault correction efforts [9]. Many different software maintenance variables are utilized as predictor variables. In this case the most common ones are complexity and size metrics, testing metrics and process quality data. A number of techniques, methods and approaches are applied to build these models. Regression analysis, neural networks, pattern recognition are just a few to name. There is an ongoing research focused on improvement and validation of existing models [13] and on finding better, more

accurate models. For example, large complex multivariate statistical models have been produced in an attempt to find a single complexity metric that will account for defects. [10].

A work has also been reported to use rule-based approaches to build prediction models in the area of software engineering [8]. A system that uses rule-based models is a subject of this thesis.

2.2.2 Rule-based Models

Rule-based modeling is most common form of computational model. Rules are generally well suited to study behavior of many different phenomena. These models receive information describing a situation, process that information using set of rules, and produce a specific response as their output. Their overall structure can be presented in the following way, Figure 1.

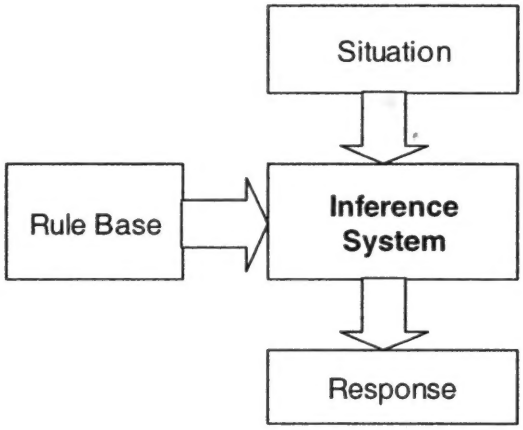

Figure 1.Structure of a Rule-Based Model

In its simples form, a rule-based model is just a set of **IF-THEN** statements called rules, which encode knowledge about phenomenon being modeled. Each rule consists of an **IF** part called the premise or antecedent (a conjunction of conditions) and a **THEN** part called the consequent or conclusion (predicted category or class). When the **IF** part is true, the rule is said to fire and the **THEN** part is asserted - it is considered to be a fact.

A set of **IF-THEN** rules is processed using Boolean logic. The expert system literature distinguishes between "forward chaining" and "backward chaining" as a method of logical reasoning. Forward chaining starts with a set of characteristics about a situation – a feature vector of independent variables – and applies these as needed until a conclusion is reached. Backward chaining, in contrast, starts with a possible conclusion – a hypothesis – and then seeks information that might validate the conclusion. Forward chaining systems are primarily data-driven, while backward chaining systems are goal-driven. In the proposed system a forward chaining method of reasoning is used.

The **IF-THEN** based model with forward chaining works in the following way: the system examines all the rule conditions (**IF**) and determines a subset, the conflict set, of the rules whose conditions are satisfied. Of this conflict set, one of those rules is triggered (fired). When the rule is fired, any actions specified in its **THEN** clause are carried out. Which rule is chosen to fire is a function of the conflict resolution strategy. Which strategy is chosen can be determined by the problem or it may be a matter of preference. In any case, it is vital as it controls which of the applicable rules are fired and thus how the entire system behaves. There are several different strategies, but here are a few of the most common:

- first applicable: is based on the fact that the rules are put in a specified order; the first applicable rule is fired;
- random: it is based on a random selection of a single rule from the conflict set; this randomly selected rule is fired;
- most specific: this category is based on the number of conditions attached to each rule; from the conflict set, the rule with the most conditions is chosen;
- least recently used: is based on the fact that each rule is accompanied by a time stamp indicating the last time it was used; a rule with the oldest time stamp is fired;
- "best" rule: this is based on the fact that each rule is given a "weight" which specifies how much it should be considered over other rules; the rule with the highest weight is fired.

Another important aspect of development of rule-based models, besides reasoning scheme, is generation of rules. There are many different methods and techniques of doing that. They can be generated on the basis of expert knowledge where rules are created by a person during interaction with a domain expert, or automatically derived from available data using a variety of different approaches and tools.

An important feature of **IF-THEN** models is that they lead to extraction of knowledge from data. This knowledge is represented in the form rules that can be read and understood by people. In a sense, these models contribute to the human knowledge about relationships, which they represent. At the same time the models can be used for prediction and estimation purposes.

2.2.3 Multiple Models

Recent years have seen an increasing interest in research in multiple models (e.g.: Decision trees, Rules sets, Neural networks, Bayesian networks and Regression). In these researches, several models of data are learned from, instead of just a single model, to produce one set of training examples [2].

In more complex combining schemes, each model produces a classification and a measure of certainty that it has in its classification. The error rate of this ensemble, called the ensemble error rate is usually compared to the error rate of the single model produced by using a single-model learning algorithm on the same training data. [2]

The task of combining multiple models can be broken down into two subtasks. The first is the generation of diverse set of base-level models. Once the based-level model models have been generated, the issue of combination of their prediction arises. [32] The next task is to combine the multiple models. Another way is to use a single model-learning algorithm with different initial settings. Multiple models can also be generated by applying a single base level learning algorithm to different versions of learning data. Different methods for manipulating the set of learning examples can be used, such as random sampling with replacement in bagging [5,32] or re-weighting misclassified training examples in boosting [11,32]. Learning multiple models can improve the accuracy and stability of single models significantly, but at the cost of losing their comprehensibility (when they possess it, as do, for example, simple decision trees and rule sets).

A lot of work has been done on using multiple models. Buntine [7] compared two model generation methods using twelve domains. In the first method, he used beam search while in the second he generated different decision trees by pruning a single learned tree in different ways. He shows that pruning a single decision tree in different ways yields trees that are all quite syntactically and functionally similar. These trees do not do as well as trees generated by beam search [2]. This agrees with the work done by Ali & Pazzani [3] on the connection between syntactic diversity of models in an ensemble and the ability of the ensemble to reduce error.

Kononenko & Kovacic [14] used three domains to compare different model generation methods produced by varying the amount of search. They compared random-start hill-climbing, stochastic-search hill-climbing and simulated annealing. Their results showed that more intensive search methods yield ensembles that have lower error rates. [2]

Kwok & Carter [15] used two domains to compare two ways of generating decision trees ensembles. In one, the user can vary the decision nodes at the root of a decision tree and in the other the user can vary decision nodes further down the tree but not at the root. They showed that varying the root leads to syntactically more dissimilar trees (which presumably are functionally more diverse). These trees produce ensembles with lower error rates. [2].

Todorovski and Dzeroski [32] combined multiple models with meta decision trees (MDTs) which specifies which model should be used to obtain a prediction instead of giving a prediction. They presented an algorithm for learning MDTs based on C4.5 algorithm for learning ordinary decision trees (ODTs). An extensive experimental evaluation of the new algorithm was performed on twenty-one datasers, combining models generated by five learning algorithms. Two algorithms for learning decision trees, a rule learning algorithm, a nearest neighbor algorithm and a naïve Bayes algorithm [34]. Their results showed that in terms of performance, MDTs combine better than voting and stacking with ODTs and are more concise than ODTs used for stacking and are thus a step towards comprehensible combination of multiple models.

2.3 Evidence Theory

2.3.1 Basic Concepts

Uncertainty is classically represented by probability functions, and diagnostic is an environment poised by uncertainty is usually handled through the application of the Bayesian theorem that permits the computation of the posterior probability over the diagnostic categories given the observed data from the prior probability over the same categories [29]. This uncertainty shows a similar solution when quantified by belief functions from the theory of Evidence.

The theory of evidence was introduced in the late Seventies by Glenn Shafer as a way of representing epistemic knowledge, starting from a sequence of seminal works of Arthur Dempster, Shafer's advisor [25]. In this formalism the best representation of chance is a belief function (b.f.) rather than a Bayesian mass distribution. They assign probability values to sets of possibilities rather than single events: their appeal rests on the fact they naturally encode evidence in favor to propositions [25]. The theory embraces the familiar idea of assigning numbers between 0 and 1 to indicate this degree of support but, instead of focusing on how this numbers are determined, it concerns the combination of degrees of belief.

The formalism provides a simple method for combining the evidence carried by a number of different sources (Dempster's rule) with no need of any a-priori distributions. In this sense, following Shafer, it can be seen as a theory of probable reasoning. A formal definition of the different levels of detail in knowledge representation is introduced, when the concept of family of compatible frames reflects the intuitive idea of different descriptions of the same phenomenon.

The most popular theory of probable reasoning is perhaps the Bayesian theory, due to the English clergyman Thomas Bayes (1702-1761), claiming that all degrees of beliefs must obey the rule of chances (i.e. the proportion of the time one of the possible outcomes of a random experiment tends to occur). Furthermore, the fourth rule of the Bayesian theory gives a precise way a Bayesian function must be updated when we learn that a certain proposition A is true:

$$P(B \mid A) = P(B \text{ and } A)/P(A); \qquad (2.1)$$

it is called Bayes' rule.

2.3.2 The Transferable Belief Model

The transferable belief model (TBM) is a model developed to represent quantified beliefs [28,27]. It covers the same domain as the Bayesian – subjectivist probabilities except it is not based on probability functions but on so-called belief functions [29]. One starts from a finite set of worlds Ω called the frame of discernment. One of its worlds denoted ω_0 corresponds to the **actual world**. The agent does not know which world in Ω corresponds to the actual world in ω_0. We have some opinion about the world that might be the actual one. For every subset of A of Ω, we can express the strength of our opinion that the actual world ω_0 belongs to A. This strength is denoted by bel (A) with bel: $2^{\Omega} \rightarrow [0,1]$. Extreme values for bel denote full belief value – 1or no belief at all with values – 0. The larger bel(A), the stronger you believe $\omega_0 \in A$. TBM is different from the classical Bayesian approach because it does not assume the additivity encountered probability theory. In probability, bel (A) = 0 implies that bel($\bar{A}$) =1 but bel(A) = bel($\bar{A}$) = 0 is possible for TBM. The additivity property that characterizes the probability functions is replaced by inequalities like:

$$Bel(A \cup B) \geq bel(A) = bel(B) - bel(A \cap B) \qquad (2.2)$$

In the TBM, one assumes that bel is a capacity monotone of order infinite, i.e., bel satisfies the following inequalities:

$$\forall\, n > 1, \forall\, A_1, A_2, \ldots A_n, \subseteq \Omega,$$

$$bel(A_1 \cup A_2 \cup \ldots A_n) \geq \sum_{i} bel(Ai) - \sum_{i>j} bel(A_i \cap A_j) \ldots - (-1)^n bel(A_1 \cap A_2 \cap \ldots A_n) \qquad (2.3)$$

These inequalities gives us eq (2.2) when n = 2.

2.3.3 Basic Belief Assignments

In order to understand above inequalities, the concept of a basic belief assignment (bba) has to be introduced. Related to *bel*, one can define its so-called Moebius transform, denoted *m* and called a basic belief assignment. Let

$$m : 2^{\Omega} \rightarrow [0,1]$$

where $m(A)$ is called the basic belief mass (bbm) given to $A \subseteq \Omega$. The value of $m(A)$ represents belief that supports A – i.e., the fact that the actual world ω_0 belongs to A without supporting any more specific subset. In the case, when ω_0 belongs to A, and nothing more is know about the value of ω_0, then some part of belief will be given to A, but no subset of A will get any positive support. In that case, $m(A) > 0$ and $m(B)=0$ for all $B \subseteq A$ and $B \neq A$.

Belief Functions

The bbm $m(A)$ does not quantify belief, denoted $bel(A)$, that the actual world ω_0 belongs to A. Indeed, the bbm m(B) given to any subset B of A also supports that ω_0 belongs to A. Hence, the belief bel(A) is obtained by summing all the bbm m(B) for $B \subseteq A$. At the end:

$$bel(A) = \sum_{\emptyset \neq B \subseteq A} m(B) \qquad \forall A \subseteq \Omega, A \neq \emptyset \qquad (2.4)$$

$$bel(\emptyset) = 0$$

Notion of belief

Following Shafer [25] we call the finite set of possibilities frame of discernment (FOD).

A basic probability assignment over a FOD θ is a function m: $2^\theta \rightarrow [0,1]$ such that,

$$M(\phi) = 0$$

and $$\sum_{A \subset \theta} m(A) = 1$$

The element of 2^θ associated to non-zero values of m are called focal elements and their union core. Now suppose a b.p.a. is introduced over an arbitrary FOD.

The belief function given the basic probability assignment, m is defined as

$$Bel(A) = \sum_{B \subset A} m(B)$$

In the theory of evidence a probability function is simply a peculiar kind of belief function.

A function Bel: $2^{\theta} \rightarrow [0,1]$ is called Bayesian belief function if

1. Bel (ϕ) = 0

2. Bel (θ) = 1

3. Bel($A \cup B$) = Bel(A) + Bel(B) whenever $A,B \subset \theta$ and $A \cap B = \phi$. A function Bel is Bayesian $\Leftrightarrow \exists$ p: $\theta \rightarrow [0,1]$ such that

$$\sum_{\theta \subset \Theta} p(\theta) = 1 \text{ and}$$

$$\text{Bel (A)} = \sum_{\theta \subset A} p(\theta)$$

for all $A \subset \theta$.

Belief functions representing distinct bodies of evidence are combined together by means of the Dempster's rule of combination:

The orthogonal sum Bel1 $\oplus$ Bel_2 of two belief functions is a function whose focal elements are all possible intersections between the combining focal elements and whose b.p.a is given by

$$m(A) = \frac{\sum_{i,j:A_i \cap B_j = A} {}^{m}1(A_i)m_2(B_j)}{1 - \sum_{i,j:A_i \cap B_j = \phi} {}^{m}1(A_i)m_2(B_j)} \qquad (2.5)$$

The orthogonal sum is easily extended to the general case of combining several belief functions.

Combination of Two Belief Functions

Suppose there are two 'distinct' pieces of evidence Ev1 and Ev2 produced by two sources of information. Let bel_1 and bel_2 be the belief functions induced by each piece of evidence. These two belief functions can be combined in at least two ways: conjunctively or disjunctively. It means that belief can be built when *both* sources of information are reliable (conjunctive combination), or when *at least one* of the two sources is reliable (disjunctive combination). The resulting belief functions $bel_{1 \wedge 2}$ and $bel_{1 \vee 2}$ are obtained from the relations:

conjunctive combination

$$m_{1\wedge 2}(A) = \sum_{X,Y\subseteq\Omega:X\cap Y=A} m_1(X) * m_2(Y) \qquad \text{for all } A \subseteq \Omega \tag{2.6}$$

disjunctive combination

$$m_{1\vee 2}(A) = \sum_{X,Y\subseteq\Omega:X\cup Y=A} m_1(X) * m_2(Y) \qquad \text{for all } A \subseteq \Omega \tag{2.7}$$

The meaning of "distinct' for two pieces of evidence has been left undefined. It lacks rigorous definition. Intuitively it means the absence of any relation. In this case the belief function bel_2 induced by the second source is not influenced by the knowledge of the belief function bel_1 induced by the first source and vice versa [30].

Decision Process within Transferable Belief Model

A theory for decision process has been fully developed within the Transferable Belief Model (TBM) [27]. Given a belief function, a probability function is generated that must be used to make decision by maximizing expected utilities. It requires first the construction of the betting frame, i.e., a list of alternatives on which the bet must be made. Let Bet_{Frame} denotes the betting frame. The granularity of Bet_{Frame} is such that if by necessity two alternatives are not distinguishable from a consequence-utility point of view, than they are pooled into the same granule.

Once the betting frame is determined. The bba m is then transformed by the so-called pignistic transformation into the pignistic probabilities *BetP*: $2^{\Omega} \rightarrow [0,1]$ with:

$$\text{BetP}(A) = \sum_{X\subseteq \text{Bet}_{\text{Frame}}, X\neq\varnothing} \frac{m(X)}{1-m(\varnothing)} \frac{\#(A\cap X)}{\#(X)} \quad \text{for all } A \subseteq \text{Bet}_{\text{Frame}}, A \neq \varnothing \tag{2.8}$$

and #(X) is the number of granules of the betting frame Bet_{Frame} in X. By construction, the pignistic probability function *BetP* is a probability function, but it is qualified as pignistic to avoid the error that would consist in considering this probability function as someone's beliefs. Someone's beliefs

are represented by *bel*, and *BetP* is just the additive measure needed to compute expected utilities when decision must be made [22].

3. Multi-Model Estimation System

A number of prediction models are used to support management tasks related to software development and maintenance activities. Many of these models are classical stimulus-response models, what means that their users do not gain any knowledge about relationships, which are represented by these models. These models are called "black box" models. Users obtain results without understanding how they have been derived. Models based on sets of **IF-THEN** rules belong to a group of so called transparent models. In such case, users do not only obtain numerical results but also learn about nature of relationships among attributes of input data. This increases understanding of modeled processes, and can lead to "wiser" changes in these processes in order to obtained desired effects.

An important aspect of prediction models is to provide user with an indication about confidence that can be assigned to the results. Users should have an indicator based on which they can formulate their own opinion how to treat the results. It is of a special importance due to the fact that prediction models have prediction rate in the range of 60% to 85% - 90%. Confidence levels associated with results and "goodness" of **IF-THEN** rules help users in better understanding of the phenomenon, model and results.

The approach proposed in the thesis addresses the issues of variety of **IF-THEN** rules and methods of their development, as well as lack of indicators about confidence levels of results obtained from **IF-THEN** based models.

3.1 Concept

There is no a single best method of generating **IF-THEN** rules from data. In this particular case it can be said that each method analyses and processes data in a different way. Application of different methods leads to different sets of **IF-THEN** rules. Each of the methods "sees" data from a different angle; therefore each method can "see" different relationships among data attributes. The aim is to take advantage of that by combining many differently derived rules into a single system.

Another important aspect is related to the concept of confidence levels that should be associated with **IF-THEN** rules. There is no certainty about rules representing relationships existing among attributes of data. In the case of prediction models, each rule predicts that given data point belongs to a given category but there are questions how good the rule is, and how much a user can believe in this prediction. There is a chance, larger or smaller, that the data point belongs to a different category than one indicated by the rule. Once confidence levels for rules are established there is a need for deriving a confidence level of the result.

A system proposed in this thesis embraces rule-based models developed using different methods and merges them based on confidence levels associated with the rules. The main idea of the approach is to:

- use a number of rule-based models developed by different methods;
- use a concept of a belief function from evidence theory [24,28] to assert confidence levels of the rules and validate them;
- use Transferable Belief Model [27] built on evidence theory to reason about belonging of a data point belongs to a specific class and confidence level of this result.

Construction details and structure of a proposed multi-model estimation system are show in Figure 2.

An originality of the concept relies on evidence theory [25].

One of the main elements of evidence theory is a concept of basic belief mass (bbm in short), which represents degree of belief that something is true. This concept is an essential element of the proposed system. A bbm value is assigned to each rule. It means that if a rule is satisfied by a given data point then there is belief equal to bbm value that this data point belongs to a category indicated by the rule. At the same time the belief of value (1-bbm) is assign to a statement that it is not known to which category the data point belongs. In other words, every rule, which is satisfied by a given data point "generates" two numbers:

- one that indicates belief that a given data point belongs to a category indicated by the rule; its value is equal to bbm;

- one that indicates that a given data point can belong to any category; its value is (1-bbm).

In this case, of course, higher the bbm value of a rule, higher belief that a given data point belongs to a category indicated by the rule, and smaller belief that a data point can belong to any category. The bbm values of all rules satisfied by a data point together with categories identified by these rules constitute the input to an inference engine. Details of this engine are presented in subsection 3.3

3.2 Model Generation Methods

IF-THEN rule-based models are developed using three different and independent techniques and tools. One set of rules is constructed by See5/C5 tool [21]. It is an updated version of the well-known C4.5 algorithm [19]. It has capability of generating classifiers that are expressed as decision trees or sets of **IF-THEN** rules. The second used tool is 4cRuleBuilder [1]. It uses supervised learning techniques to generate model of the data from discrete numerical or nominal data, and has build-in descretization schemes for continuous attributes. In overall, 4cRuleBuilder generates compact rules that use small number of selectors. The third method of generation **IF-THEN** rules is an evolutionary based approach of construction decision trees [20]. It combines genetic algorithm (GA) and genetic programming (GP) into a single algorithm. It puts emphasis on targeting two problems simultaneously – searching for the best split in attribute domains and finding the best choice of variables for splitting. Combination of GA strength to search through numeric space and GP strength to search symbolic space resulted in an approach where a process of construction of decision trees – defining splitting, selecting variables - can be performed simultaneously and "controlled" by a fitness function. Such fitness function can represent different objectives essential for a given classification process. The idea is to construct a decision tree via performing two-level optimization: parametric and structural. Once a decision three is developed a set of **IF-THEN** rules is extracted from it.

3.3 Generation and Validation of Rule Based Models

A development stage of multi-model estimation system is about construction of **IF-THEN** based models. A number of different techniques and methods can be used to generate sets of rules. Each set is the result of application of a different extraction technique. This leads to a variety of rules

that differ between one another. All sets of rules are built using the same subset of data. It is called training data and it contains data points, which are randomly selected from the original data set.

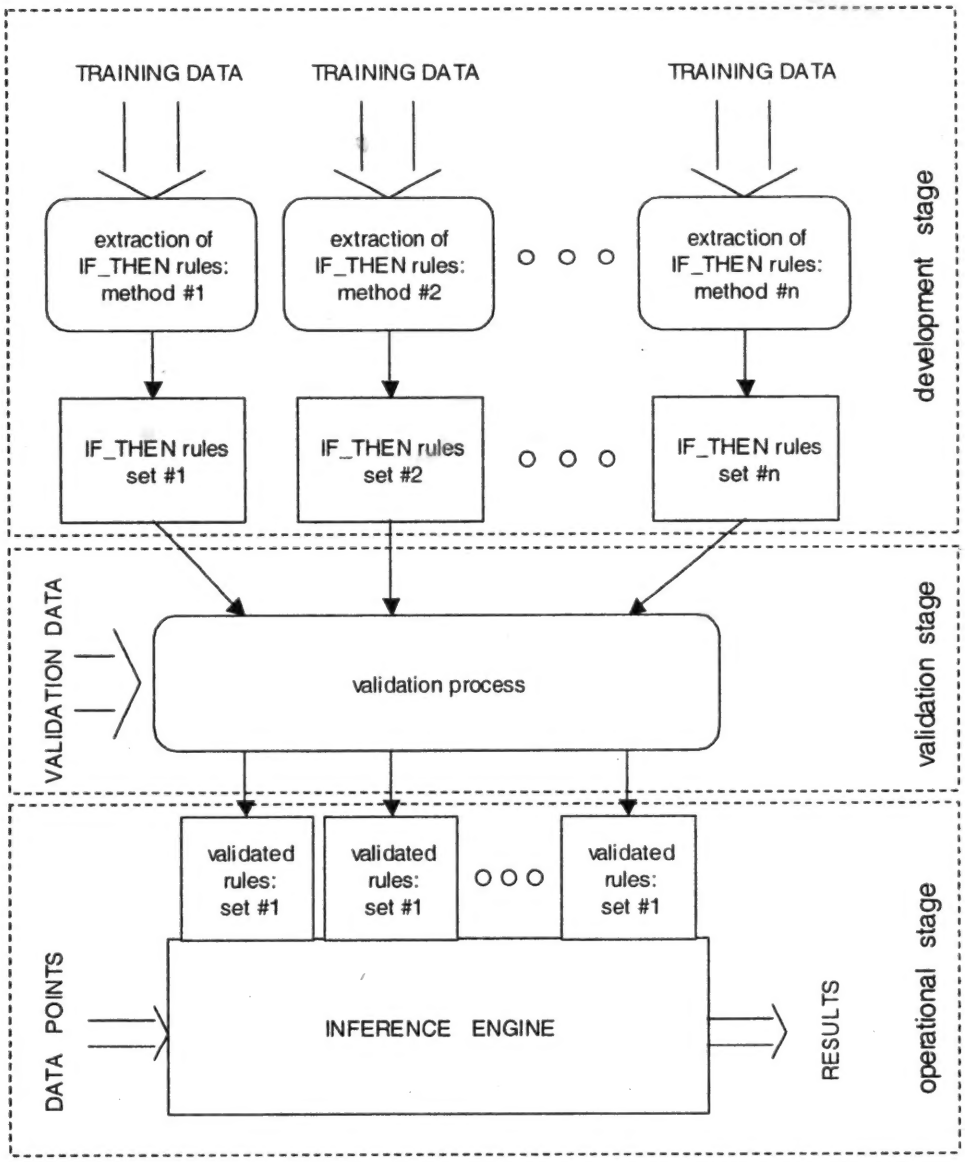

Figure 2.Multi-model Estimation System

At the time the rules are built, each rule is checked against all training data points. The number of points, which satisfy conditional part of the rule, and the number of points which satisfy both the conditional part and the consequent part of the rule are stored. These two numbers are used to generate a value. In the context of evidence theory this value is called basic belief mass – bbm, and as such is used in this thesis. It is an indicator of "goodness" of a rule. The formula used to calculate this value is shown below:

$$bbm_T = \frac{True_Positives_T + 1}{True_Positive_T + False_Positives_T + 2} \tag{3.1}$$

where $True_Positives_T$ represents a number of training data points satisfying both the conditional and the consequent parts of the rule, $False_Positive_T$, on the other hand, represents a number of testing data points satisfying only the conditional part of the rule. This formula is based on the Laplace ratio. The value calculated in that way is assigned to a rule. It represents belief in classification capabilities of this rule - it means belief that a given data point, in the case it satisfies the conditional part of the rule, can be classified as a point belonging to a class indicated by the rule. Each rule from each set has bbm value assigned to it.

All bbm values are calculated based on the training data. The same data is used to build the rules and the same data is used to evaluate their "goodness". Because of that, there is a need for more independent evaluation of rules. And this leads to a validation stage. For this purpose, the rest of the original data, left after creation of training data, is used to "check goodness" of all rules. The data is called validation data.

Each rule from each set is checked against all data points from the validation set. Once again the numbers of points that satisfy the conditional part of the rule and the number of points that satisfy both conditional and consequence parts of the rule are recorded. Another bbm value is calculated using the previously defined formula. As the results a new bbm value is obtained based on the validation data.

$$bbm_V = \frac{True_Positives_V + 1}{True_Positive_V + False_Positives_V + 2} \tag{3.2}$$

where $True_Positives_V$ represents a number of validation data points satisfying both the conditional and the consequent parts of the rule, $False_Positive_V$, on the other hand, represents a number of validation data points satisfying only the conditional part of the rule. Because the bbm values obtained from the training data should not be discarded, and the bbm values obtained from the validation data should be used – a combination of both bbm values is performed. For this purpose a special formula is proposed that preserves the value of bbm_T in the case when bbm_V is equal to 0.5 (value of 0.5 indicates 50-50 chances that something is true/false). The formula is presented below:

$$bbm_{UPDATE} = \frac{bbm_T * bbm_V}{bbm_T * bbm_V + (1 - bbm_T) * (1 - bbm_V)} \tag{3.3}$$

These updated bbm values represent beliefs that generated rules can perform proper prediction. They are used by inference engine to derive the overall result.

3.4 Inference Engine

Generation and validation of rule-based models are essential preparation steps that lead to construction of a proposed system. The system contains all generated rules together with bbm values and an inference engine. The inference engine is a core of the system. It is an implementation of the following concept. When a given data point satisfies a number of rules, then a number of bbm values that indicate belonging of this data point to the same or a number of different categories. In the case when a single category is identified by all satisfied rules, the inference engine is not engaged and prediction is finished – it is predicted that the data point belongs to the identified category. In the case when a number of different categories are identified, the Transferable Belief Model is used to derive possibilities that the data point belongs to each of these categories. A structure of the inference engine is presented in Figure 3. Behavior of the inference engine is illustrated using a simple example.

Example: Two **IF-THEN** based models have been generated to represent relationships between n-dimensional input and a single-dimensional output. There are two categories distinguished in the

output space: I and II. Each of the models contains three **IF-THEN** rules for each category. Rules and their bbm values are shown in Table 1 below.

Table 1. Rules with their bbm

Model	Output Category	IF-THEN rule	Basic Belief Masses
First	**I**	M1_C1_R1	0.60
		M1_C1_R2	**0.85**
		M1_C1_R3	**0.75**
	II	M1_C2_R1	0.67
		M1_C2_R2	0.94
		M1_C2_R3	0.80
Second	**I**	M2_C1_R1	0.80
		M2_C1_R2	0.90
		M2_C1_R3	0.67
	II	**M2_C2_R1**	**0.90**
		M2_C2_R2	.94
		M2_C2_R3	**0.85**

Let's assume that a new data point has been obtained and the system is used to predict which category this point belongs to. When checked against all rules listed above, the new data point satisfies the following set of rules: from the first model and category I – M1_C1_R2, M1_C1_R3, from the second model and category II – M2_C2_R1 and M2_C2_R3. Bbm values associated with satisfied rules are: 0.85 and 0.75 from the first model, and 0.9 and 0.85 from the second model.

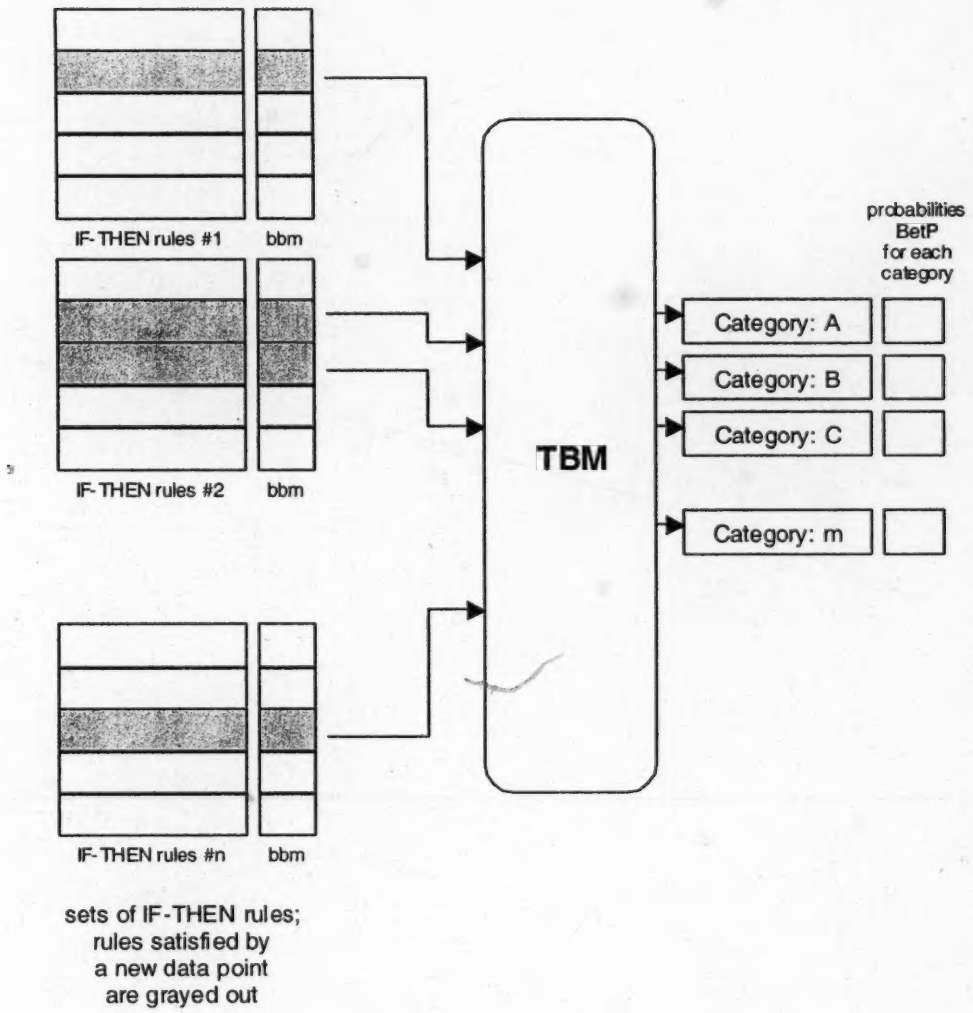

Figure 3. Structure of Inference Engine

Based on these values Table 2 is prepared. In the case of the satisfied rules from the first model the values 0.85 and 0.75 represent beliefs that the data point belongs to category I, the values 0.15 and 0.25 indicate that there is still belief that this point can belong to any category, in this case category I or category II. The same thing is done for the rules from the second model. The values in Table 2 represent input to TBM.

Table 2. Basic belief masses assigned to each possible category by satisfied rules

	M1_C1_R2	M1_C1_R3	M2_C2_R1	M2_C2_R3
0	0	0	0	0
Category I	0.85	0.75	0	0
Category II	0	0	0.9	0.85
Category: I or II	0.15	0.25	0.1	0.15

Inside TBM, the bbm values are combined using the conjunctive combination rule (equation 2.6). The results of this operation are presented in Table 3.

Table 3. Basic belief masses obtained by the conjunctive combination

x	m(x)
0	0.9481
Category I	0.0144
Category II	0.0369
Category: I or II	0.0006

The next step is to calculate two probabilities. One to represents belief that the data point belongs to category I, another that it belongs to category II. These probabilities are called pignistic. This pignistic probability BetP on the belonging of the data point to one of the two categories is computed from the bbm values shown in Table 3. Therefore according to TBM (equation 2.8):

$$BetP(\text{Category I}) = (m(\text{Category I}) + m(\text{Category: I or II})/2) / (1 - m(0))$$
$$= (0.0144 + 0.0006/2) / (1 - 0.9481) = 0.2834$$

and

$$BetP(\text{Category II}) = (m(\text{Category II}) + m(\text{Category: I or II})/2) / (1 - m(0))$$
$$= (0.0369 + 0.0006/2) / (1 - 0.9481) = 0.7166$$

To sum up:

Table 4.Pignistic probability (BetP) value

BetP (Category I)	0.2834
BetP (Category II)	0.7166

Based on values from Table 4, it can be said that in this particular case a prediction is as follows: the new data point belongs to Category I with belief of 0.2834, and to the Category II with belief of 0.7166.

PART II: KNOWLEDGE EXTRACTION

4 UCI Machine Learning Repository Data Set

This is a repository of databases [33], domain theories and data generators that are used by the machine learning community for the empirical analysis of machine learning algorithms. The repository presented the following characteristics: Database Name; Number of Instances (i.e. examples, data points, observations); Number of Features (i.e. dimensions, attributes); Number of Classes (assuming a discrete class variable); Percent of Features that have continuous/integer values; Percent of Features that have nominal values; Missing Features (yes or no); Highest Reported Accuracy (taken from "Past Usage"); Percent of Instances in the Majority Class.

4.1 Heart Disease Data (Cleveland Clinic Foundation)

The heart disease database was collected by Robert Detrano from V.A. Medical Center, Long Beach and Cleveland Clinic Foundation. The data concerns diagnosis of heart disease. All attributes are numeric-valued. The database contains 13 attributes (which have been extracted from a larger set of 75). Table 5 below shows the 13 attributes and the predicted class attribute, their values and some information about them.

Table 5.Heart Disease Attributes, Values and Information

Attributes	Values and Description
Class (num)	Absent (<50% diameter narrowing) Present (>50% diameter narrowing)
Age	In years
Sex	1 = male; 2 = female
Cp (Chest pain type)	Typical angina, atypical angina, non-anginal pain, asymptomatic
Trestbps (Resting blood pressure)	In mm Hg on admission to the hospital
Chol	Serum cholestoral in mg/dl
Fbs	Fasting blood sugar > 120 mg/dl (1 = true; 0 = false)
Restecg	Resting electrocardiographic results (Values 0,1,2)
Thalach	Maximum heart rate achieved
Exang	Exercise induced angina (1 = yes; 0 = no)

oldpeak	ST depression induced by exercise relative to rest
Slope	The slope of the peak exercise ST segment
Ca	Number of major vessels (0-3) colored by flourosopy
Thal	3 = normal; 6 = fixed defect; 7 = reversable defect

4.2 Lymphography Data

This lymphography domain was obtained from the University Medical Centre, Institute of Oncology, Ljubljana, Yugoslavia by M. Zwitter and M. Soklic. It contains 148 instances; 19 attributes including the class; 4 classes; with no missing data values. All the attributes values in the database have been entered as numeric values corresponding to their index in the list of attribute values for that attribute domain as given in Table 6 below.

Table 6.Lymphography Data Attributes and Values

Attributes	**Values**
Class	Normal find(2), metastases(81), malign lymph(61), fibrosis(4)
Lymphatics	Normal, arched, deformed, displaced
Block of affere	no, yes
Bl. of lymph. c	no, yes
Bl. of lymph. s	no, yes
By pass	no, yes
Extravasates	no, yes
Regeneration of	no, yes
Early uptake in	no, yes
Lym.nodes dimin	0-3
Lym.nodes enlar	1-4
Changes in lym	bean, oval, round
Defect in node	no, lacunar, lac. marginal, lac. central
Changes in node	no, lacunar, lac. margin, lac. central
Changes in stru	no, grainy, drop-like, coarse, diluted, reticular, stripped, faint
Special forms	no, chalices, vesicles
Dislocation of	no, yes
Exclusion of no	no, yes
No. of nodes in	0-9, 10-19, 20-29, 30-39, 40-49, 50-59, 60-69, >=70

4.3 Breast Cancer Data

This breast cancer domain was obtained from the University Medical Centre, Institute of Oncology, Ljubljana, Yugoslavia. Thanks go to M. Zwitter and M. Soklic for providing the data. It contains 286 instances, 2 classes, 9 attributes plus the class attribute. This data set includes 201 instances of one class and 85 instances of another class. The instances are described by 9 attributes, some of which are linear and some are nominal. The data set contains 9 missing values – node caps (8) and breast (1).

Table 7. Breast Cancer Attributes and Values

Attributes	Values
Class	no-recurrence-events (201 instances), recurrence-events (85 instances)
Age	10-19, 20-29, 30-39, 40-49, 50-59, 60-69, 70-79, 80-89, 90-99.
Menopause	lt40, ge40, premeno.
Tumor-size	0-4, 5-9, 10-14, 15-19, 20-24, 25-29, 30-34, 35-39, 40-44, 45-49, 50-54, 55-59.
Inv-nodes	0-2, 3-5, 6-8, 9-11, 12-14, 15-17, 18-20, 21-23, 24-26, 27-29, 30-32, 33-35, 36-39.
Node-caps	yes, no.
Deg-malig	1, 2, 3.
Breast	left, right.
Breast-quad	left-up, left-low, right-up, right-low, central.
Iradiat	yes, no.

5 Application of Multi-Model Estimation System to ML Data

5.1 Breast Cancer Data

5.1.1 Model Building

The C5 method has led to extraction of eight rules: three for "*no recurrence events*", and five for "*recurrence events*". A prediction model built on these rules gives 80.4% correct classifications for the training data. Due to space constraints only two of the most important rules generated for each output category – *no recurrence, recurrence* – are presented below. For each rule, a confidence level (bbm) is shown.

Rule B_C5_ c1_1: **IF** *invasive nodes* is of the range *<0, 2>*
THEN there are *no recurrence events*
Confidence level (bbm_T): 0.806

Rule B_C5_ c1_2: **IF** *breast quadrant* is *left_up*
THEN there are *no recurrence events*
Confidence level (bbm_T): 0.750

Rule B_C5_ c2_1: **IF** *invasive nodes* is of the range *<6, 8>*
AND there is *irradiation*
THEN there are *recurrence events*
Confidence level (bbm_T): 0.833

Rule B_C5_ c2_2: **IF** *invasive nodes* is of the range *<9, 11>*
THEN there are *recurrence events*
Confidence level (bbm_T): 0.833

The application of 4C method has resulted in fourteen rules for "*no recurrence events*", and fifteen for "*recurrence events*". In this case the classification rate for the training data is 83.6%. As above, only two rules for each category are presented:

Rule B_4C_ c1_1: **IF** *age* is of the range *<20, 29>* **OR** *<50, 79>*
AND *tumor size (mm)* is of the range *<0, 19>* **OR** *<30, 44>*
OR *<50, 54>*
AND *invasive nodes* is of the range *<0, 2>* OR *<6, 8>* **OR** *<24, 26>*
AND *degree of malignancy* is *1* **OR** *2*
AND there is *no irradiation*

THEN there are *no recurrence events*

Confidence level (bbm_T): 0.974

Rule B_4C_ c1_12: **IF** *age* is of the range *<20, 29>* **OR** *<40, 49>* **OR** *<70, 79>*
AND *menopause is at Premeno (if patient is at pre-menopause stage)* **OR** *lt40*
AND *invasive nodes* is of the range *<0, 2>* **OR** *<24, 26>*
THEN there are *no recurrence events*

Confidence level (bbm_T): 0.950

Rule B_4C_ c2_1: **IF** *age* is of the range *<20, 29>* **OR** *<40, 59>*
AND *tumor size (mm)* is of the range *<15, 19>* **OR** *<25, 34>*
OR *<40, 44>*
AND *degree of malignancy* is *3*
AND *breast is left*
AND *breast quadrant* is *left_low* **OR** *right_up* **OR** *right_low*
THEN there are *recurrence events*

Confidence level (bbm_T): 0.923

Rule B_4C_ c2_3: **IF** *age* is of the range *<20, 29>* **OR** *<30, 39>* **OR** *<50, 59>*
AND *menopause is at Premeno (if patient is at pre-menopause stage)*
AND *tumor size (mm)* is of the range *<25, 29>* **OR** *<35, 39>*
AND *degree of malignancy* is 1 **OR** 3
THEN there are *recurrence events*

Confidence level (bbm_T): 0.900

The genetic-based approach has extracted a total of twenty-four rules: fourteen for "*no recurrence events* " and ten for "*recurrence events*". The classification rate, for the training data, is 84.7%. The rules are:

Rule B_GB_ c1_1: **IF** there is *no irradiation*
AND *invasive nodes* is of the range *<0, 2>* **OR** <24, 26>
THEN there are *no recurrence events*

Confidence level (bbm_T): 0.836

Rule B_GB_ c1_7: **IF** there is *irradiation*
AND *tumor size (mm)* is of the range *<5, 9>* **OR** *<15, 19>*
AND *invasive nodes* is of the range *<0, 2>* **OR** <24, 26>
THEN there are *no recurrence events*

Confidence level (bbm_T): 0.750

Rule B_GB_ c1_12: **IF** there is *irradiation*
AND *tumor size (mm)* is of the range *<25, 29>*
AND *breast quadrant* is *left_up*
THEN there are *no recurrence events*
Confidence level (bbm_T): 0.750

Rule B_GB_ c2_6: **IF** there is *no irradiation*
AND *invasive nodes* is of the range *<9, 11>*
THEN there are *recurrence events*
Confidence level (bbm_T): 0.800

Rule B_GB_ c2_7: **IF** there is *irradiation*
AND *tumor size (mm)* is of the range *<30, 34>*
THEN there are *recurrence events*
Confidence level (bbm_T): 0.769

5.1.2 Validation of Rules

Using the extracted rules, three prediction models have been built: one based on C5 generated rules, one based on 4C generated rules, and one based on rules extracted by the evolution-based approach. All rules used for model building have associated with them confidence levels. These levels represent "goodness" of the rules prediction capabilities. According to the proposed concept, the next step in building a system is to validate all rules. In order to do this the second set of data, called a validation set, is used. All the previously generated rules are checked against data points from the validation set. Based on number of $True_Positive_V$ and $False_Positive_V$ data points a new belief value bbm_V is calculated for each rule. The bbm_V values are used to update original confidence levels. As the result updated belief values, bbm_{update}, are created. The effect of this process is presented in Table 8, Table 9 and Table 10.

The best rules generated by C5 for category "*no recurrence events*" have kept their high values, Table 8. In the case of rules B_C5_c1_1 and B_C5_c2_2, their bbm values increased after the validation phase (there are some cases where bbm values are not changed – it means that a rule has not been satisfied by any validation data point).

For category "*recurrence events*", on the other hand, the bbm value of one of the best rules B_C5_c2_1, had decreased. In such case, a different rule B_C5_c2_3, has higher value of bbm and replaced the rule B_C5_c2_1 in the set of best rules.

Table 8.Original and updated values of bbm for rules generated by C5

Rule Number	Original (bba_T)	After Validation (bba_{update})
Category: no recurrence events		
B_C5_ c1_1	**0.806**	**0.935**
B_C5_ c1_2	**0.750**	**0.878**
B_C5_ c1_3	0.500	0.500
Category: recurrence events		
B_C5_ c2_1	**0.833**	0.714
B_C5_ c2_2	**0.833**	**0.833**
B_C5_ c2_3	0.800	**0.800**
B_C5_ c2_4	0.600	0.600
B_C5_ c2_5	0.545	0.615

A new rule is presented below:

Rule B_C5_ c2_3: **IF** *menopause is at Premeno (if patient is at pre-menopause stage)*
AND *invasive nodes* is of the range *<15, 17>*
THEN there are *recurrence events*
Confidence level (bbm_{UPDATE}): 0.800

The validation process has been also performed for rules generated by 4C, Table 9. In the category "*no recurrence events*" all but the rule B_4C_c1_5, have increased bbm values. It has changed the second best rule in the category. Instead of the rule B_4C_c1_12, the second best rule is B_4C_c1_10. This rule is presented below.

Table 9.Original and updated values of bbm for rules generated by 4C

Rule Number	Original (bba_T)	After Validation (bba_{update})
Category: no recurrence events		
B_4C_ c1_1	**0.974**	**0.995**
B_4C_ c1_2	0.923	0.977
B_4C_ c1_3	0.939	0.979
B_4C_ c1_4	0.917	0.957
B_4C_ c1_5	0.917	0.880
B_4C_ c1_6	0.909	0.968
B_4C_ c1_7	0.923	0.941
B_4C_ c1_8	0.929	0.963
B_4C_ c1_9	0.917	0.957
B_4C_ c1_10	0.889	**0.980**
B_4C_ c1_11	0.900	0.947
B_4C_ c1_12	**0.950**	0.971
B_4C_ c1_13	0.923	0.960
B_4C_ c1_14	0.857	0.857
Category: recurrence events		
B_4C_ c2_1	**0.923**	0.857
B_4C_ c2_2	0.800	0.800
B_4C_ c2_3	**0.900**	**0.900**
B_4C_ c2_4	0.667	0.500
B_4C_ c2_5	0.833	**0.909**
B_4C_ c2_6	0.875	0.778
B_4C_ c2_7	0.833	0.833
B_4C_ c2_8	0.800	0.800
B_4C_ c2_9	0.833	0.833
B_4C_ c2_10	0.750	0.750
B_4C_ c2_11	0.833	0.833
B_4C_ c2_12	0.667	0.667
B_4C_ c2_13	0.667	0.667
B_4C_ c2_14	0.667	0.667
B_4C_ c2_15	0.667	0.667

Rule B_4C_ c1_10: **IF** *age* is of the range *<20, 29>* **OR** *<40, 49>* **OR** *<60, 79>*
AND *tumor size (mm)* is of the range *<0, 14>* **OR** *<25, 29>*
OR *<40, 44>*

AND *invasive nodes* is of the range *<0, 2>* **OR** *<15, 17>* **OR** *<24, 26>*
AND *breast is right*
THEN there are *no recurrence events*

Confidence level (bbm_{UPDATE}): 0.980

The rules of the second category "*recurrence events*" also have their bbm values changed. This has lead to replacement of the best rule. The rule B_4C_c2_1 is replaced by the rule B_4C_c2_5. The new rule is:

Rule B_4C_ c2_5: **IF** *age* is of the range *<20, 29>* **OR** *<50, 59>*
AND *menopause is at Premeno (if patient is at pre-menopause stage)*
AND *tumor size (mm)* is of the range *<25, 34>*
AND *invasive nodes* is of the range *<0, 2>* **OR** *<9, 11>* **OR** *<24, 26>*
AND *there is no node caps*
AND *breast quadrant* is *left_up* **OR** *right_up*
THEN there are *recurrence events*

Confidence level (bbm_{UPDATE}): 0.909

The validation process performed on genetic-based generated rules has brought a change in the second category "*recurrence events*"; see Table 10. The rule B_GB_c2_6 and B_GB_c2_7 have been replaced by B_GB_c2_2, B_GB_c2_3 and B_GB_c2_10. The bba values of rules from the first category "*no recurrence events*" have improved and stayed the highest. New rules in the second category are shown below.

Table 10.Original and updated values of bbm for rules generated by GB

Rule Number	Original (bba_T)	After Validation (bba_{update})
Category: no recurrence events		
B_GB_ c1_1	**0.836**	**0.953**
B_GB_ c1_2	0.750	0.750
B_GB_ c1_3	0.750	0.500
B_GB_ c1_4	0.500	0.500
B_GB_ c1_5	0.667	0.667
B_GB_ c1_6	0.750	0.750
B_GB_ c1_7	**0.750**	**0.857**
B_GB_ c1_8	0.667	0.667
B_GB_ c1_9	0.667	0.500

B_GB_ c1_10	0.667	0.667
B_GB_ c1_11	0.750	0.750
B_GB_ c1_12	**0.750**	**0.857**
B_GB_ c1_13	0.750	0.750
B_GB_ c1_14	0.667	0.667
Category: recurrence events		
B_GB_ c2_1	0.667	0.667
B_GB_ c2_2	0.750	**0.857**
B_GB_ c2_3	0.750	**0.857**
B_GB_ c2_4	0.750	0.600
B_GB_ c2_5	0.667	0.667
B_GB_ c2_6	**0.800**	0.800
B_GB_ c2_7	**0.769**	0.526
B_GB_ c2_8	0.500	0.500
B_GB_ c2_9	0.600	0.429
B_GB_ c2_10	0.750	**0.857**

Rule B_GB_ c2_2: **IF** there is *irradiation*
AND *tumor size (mm)* is of the range *<20, 24>*
AND *invasive nodes* is of the range *<3, 5>* **OR** <15, 17>
AND *breast quadrant* is *left_low*
THEN there are *recurrence events*
Confidence level (bbm_{UPDATE}): 0.857

Rule B_GB_ c2_3: **IF** there is *irradiation*
AND *tumor size (mm)* is of the range *<25, 29>*
AND *breast quadrant* is *right_low* **OR** *central*
THEN there are *recurrence events*
Confidence level (bbm_{UPDATE}): 0.857

Rule B_GB_ c2_10: **IF** there is *no irradiation*
AND *invasive nodes* is of the range *<3, 5>* **OR** <15, 17>
AND *breast quadrant* is *right_up*
THEN there are *recurrence events*
Confidence level (bbm_{UPDATE}): 0.857

5.1.3 Analysis of IF-THEN Rules

Development and validation stages result in three sets of **IF-THEN** rules (each set represents a single model) and bbm values associated with them. Let's take a closer look at the ones, which have the highest values of bbm in each set in each category (marked by bold fonts in Table 8, Table 9 and Table 10 in "After Validation" column). For category "*no recurrence events*" the range of bbm values is from 0.878 to 0.995. The rules with highest bbm values are generated by 4C: B_4C_ c1_1 with bbm of 0.995, and B_4C_ c1_10 with bbm of 0.980. The rules B_GB_ c1_7 and B_GB_ c1_12, generated by genetic-based approach, have the lowest bbm values among three sets.

An interesting observation can be made when all these rules, from category "*no recurrence events*", are compared. Rules B_C5_c1_1, B_4C_ c1_1, B_4C_c1_3 and B_GB_ c1_1 are similar. The most generic one seems to be the rule generated by C5 method. It contains a single attribute "*invasive nodes*" values of which have to be in the range <0,2>. The rule B_GB_ c1_1 adds additional range of values for the attribute "*invasive nodes*", it means <24, 26>. At the same time the rule has one more restriction. The data points classified as "*no recurrence events*" should have the value "*no irradiation*". The rules generated by 4C method are the strictest ones. They put additional restrictions regarding the attributes "*age*", "*tumour size*", "*degree of malignancy*" and "*breast quadrant*". These four rules look like a series of more and more specialized rules. The core of all these rules is the condition "*invasive nodes* is of the range *<0, 2>*". The highest belief value of 0.995 is with the rule B_4C_ c1_1. Therefore, one can conclude that:

IF *age* is of the range *<20, 29>* **OR** *<50, 79>*
AND *tumor size (mm)* is of the range *<0, 19>* **OR** *<30, 44>* **OR** *<50, 54>*
AND *invasive nodes* is of the range *<0, 2>* **OR** *<6, 8>* **OR** *<24, 26>*
AND *degree of malignancy* is *1* **OR** *2*
AND there is *no irradiation*
THEN there are *no recurrence events*

In the case of the category "*recurrence events*", the rules have smaller values of bbm in a range from 0.800 up to 0.909. This indicates that it is difficult to find a dominating rule indicating the category "*recurrence events*". Inspection of the rules points only to a single attribute "*invasive nodes*" which occurs in 5 out of 7 best rules for the second category. However, because of the fact that this attribute is also common in the rules of the first category, it seems that more information is needed

to properly classify the data points. Again the 4C-generated rule B_4C_c2_5 has the highest value of bbm equal to 0.909. This rule is:

IF *age* is of the range *<20, 29>* **OR** *<50, 59>*
AND *tumor size (mm)* is of the range *<25, 34>*
AND *invasive nodes* is of the range *<0, 2>* **OR** *<9, 11>* **OR** *<24, 26>*
AND *menopause is at Premeno (if patient is at pre-menopause stage)*
AND *there is no node caps*
AND *breast quadrant* is *left_up* **OR** *right_up*
THEN there are *recurrence events*

It looks like addition of information about such attributes as "*degree of malignancy*", "*irradiation*", "*node caps*" and "*breast quadrant*" leads to better classification process.

5.1.4 Classification of Breast Cancer Cases

The constructed multi-model estimation system has been used to classify cases related to breast cancer in the sense of recurrence of events. Two aspects of the results are important – classification rate and confidence levels assigned to obtained classifications.

The testing set contains 46 data points. Using only a single set of rules generated by C5, 4C and GB the classification rates are 54.35%, 54.35% and 67.39% respectively. Application of the multi-model estimation system increased classification rate to 71.74% (it is translated to thirty-three proper classifications, comparing to thirty-one obtained for GB rules). Out of all testing data points, twenty data points satisfy rules that lead to contradicting classification. The rest of data points, twenty-six in total, have satisfied rules that uniquely identify point's belonging to a single category (most data points have satisfied at least a single rule from each model, however some points have been satisfied by rules coming from more than one models). In all these cases probability BetP that a given data point belongs to a single category is very high 0.99. More interesting is the case of twenty data points that have lead to non-unanimous classification results. For the five most interesting cases, the rules are presented in Table 11.

Table 11.Rules satisfied by the five "confusing" data points

	C5		4C		GB		Original Category	Predicted Category
	I	II	I	II	I	II		
1		c2_1			c1_8		II	II
2		c2_4			c1_2		I	I
3			c1_6			c2_9	I	I
4		c2_4			c1_4		I	II
5		c2_4	c1_6			c2_10	II	I

It can be easily observed that each of these points classified to different categories. In the case of points 1, 2, 3 and 4, each point satisfies two rules. The values of beliefs assigned to these rules have significant influence on the outcome. For the case of the point 1, the rule B_C5_c2_1 has bbm of 0.714 and the rule B_GB_c1_8 has bbm of 0.667. These values of beliefs applied with TBM lead to BetP values indicated belonging of the point to the second category, Table 12. However, it has to be said that the BetP values, 0.455 and 0.545, indicate that the data point is "located" on the border between two categories. Similar situation is true for point number 2. In the case of the point number 3, the higher value of belief of the rule B_4C_c1_6 (0.968) leads to the high confidence in the classification outcome. The fact that bbm value of the rule B_C5_c2_4 is higher than bbm of B_GB_c1_4, and the bbm of B_4C_c1_6 is higher than bbms of B_C5_c2_4 and B_GB_c2_10, has led to misclassification of data points 4 and 5.

Table 12.BetP values for the seven "confusing" data points

	Data Points				
	1	2	3	4	5
BetP (Category I)	0.455	0.625	0.961	0.417	0.633
BetP (Category II)	0.545	0.375	0.039	0.583	0.367

5.2 Lymphography Data

5.2.1 Model Building

C5 method has been used first. Nine rules have been generated: one for "*a normal find*", three for "*metastases*", three for "*malign lymph*" and two for "*fibrosis*". A model generated by C5 gives 88.8% correct classifications for the training data. Once again two the most important rules generated for each output category are presented below.

Rule L_C5_ c1_1: **IF** are no *changes in node*
THEN there is a *normal find*
Confidence level (bbm_T): 0.750

Rule L_C5_ c2_1: **IF** there is no *regeneration of lymph*
AND there is no *early uptake in*
AND there is *exclusion of no*
THEN there are *metastases*
Confidence level (bbm_T): 0.895

Rule L_C5_ c3_1: **IF** *changes in node* is *lacunar central*
THEN there is *malign lymph*
Confidence level (bbm_T): 0.900

Rule L_C5_ c4_1: **IF** *lymph node* diminishes by 3
THEN there is *fibrosis*
Confidence level (bbm_T): 0.750

4C, on the other hand, has generated twelve rules – one for "*a normal find*", five for "*metastases*", five for "*malign lymph*" and one for "*fibrosis*". The classification rate is 85.7% for the training data. As above, only two rules for each case are presented:

Rule L_4C_ c1_1: **IF** *lymphatic* is *normal*
THEN there is *normal find*
Confidence level (bbm_T): 0.750

Rule L_4C_ c1_2: **IF** there is *block of affere*
AND *change in node* is *lacunar* marginal
AND *changes in structure* is *grainy* **OR** *drop like* **OR** *coarse*

OR *diluted* **OR** *recticular* **OR** *faint*
AND *number of nodes in lymph* is of the range <0,29>
THEN there are *metastases*

Confidence level (bbm_T): 0.972

Rule L_4C_ c3_1: **IF** there is *early uptake in*
AND *changes in node* is *lacunar* **OR** *lacunar central*
AND there are *special forms of vesicles*
AND *number of nodes in lymph* is of the range <10,59> **OR** <>=70>
THEN there is *malign lymph*

Confidence level (bbm_T): 0.964

Rule L_4C_ c4_1: **IF** there is a *by pass*
AND *lymph node* diminishes by 2 **OR** 3
THEN there is *fibrosis*

Confidence level (bbm_T): 0.800

Genetic-based approach has lead to seventeen rules: eight for "*metastases*", six for "*malign lymph*" and three for "*fibrosis*". Genetic approach has not generated any rules for "*a normal find*". Classification rate for the training is 91.4%. The rules are:

Rule L_GB_ c2_6: **IF** there are *special forms of chalices*
AND there is no *early uptake in*
THEN there are *metastases*

Confidence level (bbm_T): 0.909

Rule L_GB_ c3_4: **IF** there are *special forms of vesicles*
AND there is *early uptake in*
AND there is no *change in node* or it is *lacunar central*
THEN there is *malign lymph*

Confidence level (bbm_T): 0.889

Rule L_GB_ c4_2: **IF** there are *special forms of vesicles*
AND there is no *early uptake in*
AND *lymph nodes* diminishes by 3
THEN there is *fibrosis*

Confidence level (bbm_T): 0.667

5.2.2 Validation of Rules

Validation of rules generated by C5 has brought two changes, Table 13. A rule L_C5_c2_1 has been replaced by the rule L_C5_c2_2, which new updated value of bbm increased and suppress the updated value of bbm of the rule L_C5_c2_1. In a very similar way, the rule L_C5_c3_2 has replaced the best rule L_C5_c3_1 in the category "*malign lymph*". Two new rules are shown.

Table 13.Original and updated values of bbm for rules generated by C5

Rule Number	Original (bba_T)	After Validation (bba_{update})
Category: a normal find		
L_C5_ c1_1	**0.750**	**0.600**
Category: metastases		
L_C5_ c2_1	**0.895**	0.895
L_C5_ c2_2	0.854	**0.913**
L_C5_ c2_3	0.800	0.800
Category: malign lymph		
L_C5_ c3_1	**0.900**	0.931
L_C5_ c3_2	0.824	**0.933**
L_C5_ c3_3	0.556	0.294
Category: fibrosis		
L_C5_ c4_1	**0.750**	**0.857**
L_C5_ c4_2	0.667	0.667

Rule L_C5_ c2_2: **IF** *change in node* is *lacunar central*
THEN there is *malign lymph*
Confidence level (bbm_{UPDATE}): 0.913

Rule L_C5_ c3_2: **IF** there is no *regeneration of lymph*
AND there is *early uptake in*
AND *change in node* is *lacunar*
THEN there is *malign lymph*
Confidence level (bbm_{UPDATE}): 0.933

In the case of rules generated by 4C method, the rules recognized as the best ones during extraction of rules have stayed as the best rules also after validation, Table 14.

Table 14.Original and updated values of bbm for rules generated by 4C

Rule Number	Original (bba_T)	After Validation (bba_{update})
Category: a normal find		
L_4C_ c1_1	**0.750**	**0.750**
Category: metastases		
L_4C_ c2_1	**0.972**	**0.995**
L_4C_ c2_2	0.938	0.938
L_4C_ c2_3	0.889	0.960
L_4C_ c2_4	0.857	0.857
L_4C_ c2_5	0.875	0.875
Category: malign lymph		
L_4C_ c3_1	**0.964**	**0.994**
L_4C_ c3_2	0.900	0.947
L_4C_ c3_3	0.947	0.980
L_4C_ c3_4	0.750	0.750
L_4C_ c3_5	0.750	0.857
Category: fibrosis		
L_4C_ c4_1	**0.800**	**0.889**

Validation of the original set of rules generated by genetic-based method has also brought a single change. In the case of category "*malign lymph*", an updated bbm value of the rule L_GB_c3_4 has decreased. At the same time a new bbm value of the rule L_GB_c3_3 has increased from 0.867 to 0.975 what has pushed this rule to the set of the best rules. The rule L_GB_c3_3 is presented.

Table 15.Original and updated values of bbm for rules generated by GB

Rule Number	Original (bba_T)	After Validation (bba_{update})
Category: a normal find		
Category: metastases		
L_GB_ c2_1	0.667	0.667
L_GB_ c2_2	0.667	0.667
L_GB_ c2_3	0.857	0.923
L_GB_ c2_4	0.667	0.800
L_GB_ c2_5	0.909	0.909
L_GB_ c2_6	**0.909**	**0.930**

L_GB_ c2_7	0.846	0.943
L_GB_ c2_8	0.875	0.875
Category: malign lymph		
L_GB_ c3_1	0.600	0.600
L_GB_ c3_2	0.667	0.667
L_GB_ c3_3	0.867	**0.975**
L_GB_ c3_4	**0.889**	0.941
L_GB_ c3_5	0.714	0.833
L_GB_ c3_6	0.500	0.500
Category: fibrosis		
L_GB_ c4_1	0.667	0.667
L_GB_ c4_2	**0.667**	**0.800**
L_GB_ c4_3	0.667	0.667

Rule L_GB_ c3_3: **IF** there are *special forms of vesicles*
AND there is *early uptake in*
AND *change in node* is *lacunar*
THEN there is *malign lymph*

Confidence level (bbm_{UPDATE}): 0.975

5.2.3 Analysis of IF-THEN Rules

As in the case of the previous experiment, let's examine the best IF-THEN rules (marked by bold fonts in Table 13, Table 14 and Table 15 in "After Validation" column), which have been extracted to find relationships between data attributes and different problems with the lymph nodes.

For the category "*a normal find*" there are only two rules – one extracted by C5 methods and the other one extracted by 4C method. Both of them have relatively small value of bbm values: 0.600 for the C5-generated rule and 0.750 for the 4C-generated rule. Both rules do not have any overlapping regarding attributes they recognize as important.

In the category "*metastases*", all three rules generated by C5, 4C and genetic-based method are different. Also in this case there is no much overlapping. However, bbm values assigned to each rule are above 0.900.

The similarities among best rules appear in the case of the third category "*malign lymph*". The two condition parts that appear in each rule are: "there is *early uptake in*" and "*change in node* is *lacunar*". Additionally, rules L_4C_c3_1 and L_GB_c3_3 have a common condition: "there are *special forms of vesicles*". Presented three conditions constitute the rule L_GB_c3_3 with confidence level of 0.975. This rule, presented one more time below, can be suggested as a practical rule for classification of "*malign lymph*" cases.

IF there are *special forms of vesicles*
AND there is *early uptake in*
AND *change in node* is *lacunar*
THEN there is *malign lymph*

The last category "*fibrosis*" is represented by three rules. All of them have bbm values in the range from 0.800 to 0.889. They have a common condition: "*lymph node diminishes by 3*". The simplest rule, L_C5_c4_c1, has only this condition. The other two rules, L_4C_c4_1 and L_GB_c4_2, have additional attributes "*by pass*" and "there are *special forms of vesicles*", "there is *early uptake in*" respectively.

5.2.4 Prediction of Lymph Nodes Problems

Prediction of different problems associated with lymph nodes has been preformed using the multi-model estimation system. Classification rate and confidence levels assigned to obtained classifications have been recorded.

The testing set contains 24 data points. Using only a single set of rules generated by C5, 4C and GB the classification rates are 87.5%, 75.0% and 91.67% respectively (as it has been mentioned before some data points have not satisfied any rules). Application of the multi-model estimation system increased classification rate to 95.83% (it is translated to twenty-three proper classifications, comparing to twenty-two obtained for GB rules). In the case of effort prediction, three data points satisfy rules that indicate a contradicting classification. Twenty-one remaining data points are unanimously classified by all rules that are part of the system. Confidence levels of these classifications are around 0.99. The rules that are satisfied for these three points are presented in Table 16.

Table 16.Rules satisfied by the three "confusing" data points

	C5				4C				DT				Original Category	Predicted Category
	I	II	III	IV	I	II	III	IV	I	II	III	IV		
1		c2_2					c3_3 c3_4				c3_5		III	III
2		c2_2				c2_2					c3_5		II	II
3		c2_1 c2_2					c3_3			c2_6			II	II

All three points are classified to different categories by rules extracted by different method. Looking at BetP values, Table 17, one can say that all these cases are not too complicated. The values of BetP indicate clear classifications of the points done by TBM.

Table 17.BetP values for the three data points

	Data Points		
	1	2	3
BetP (Category I)	0.000	0.001	0.000
BetP (Category II	0.009	0.970	0.970
BetP (Category III)	0.901	0.028	0.030
BetP (Category IV)	0.000	0.001	0.000

5.3 Heart Disease Data

5.3.1 Model Building

The method that has been used as the first is C5. As the result fourteen rules have been generated: six for "*heart disease is absent*", and eight for "*heart disease is present*". A model generated by C5 gives 92.9% correct classifications for the training data. Due to space constrains only two the most important rules generated for each output category – no recurrence, recurrence – are presented below. For each rule, a confidence level (bbm) is shown.

Rule H_C5_ c1_1:**IF** *age* is of the range <26,45>
AND no *major vessels* is colored by *flourosopy*
AND *thal* is *normal*

THEN *heart disease* is *absent (<50% diameter narrowing)*

Confidence level (bbm_T): 0.962

Rule H_C5_ c1_2: **IF** the type of *Chest Pain* is *non-anginal pain*
AND *ST depression* (induced by exercise relative to rest) is of the range <-1.0,0.5>
THEN *heart disease* is *absent (<50% diameter narrowing)*

Confidence level (bbm_T): 0.935

Rule H_C5_ c2_1: **IF** sex is male
AND the type of *Chest Pain* is *asymptomatic*
AND 1 *major vessel* is colored by *flourosopy*
THEN *heart disease* is *present (>50% diameter narrowing)*

Confidence level (bbm_T): 0.952

Rule H_C5_ c2_2: **IF** *resting electrocardiographic results* shows probable
OR definite left ventricular hypertrophy by Estes' criteria
AND 2 *major vessels* are colored by *flourosopy*
THEN *heart disease* is *present (>50% diameter narrowing)*

Confidence level (bbm_T): 0.938

4C, on the other hand, has generated six rules for "*heart disease is absent*", and seven for "*heart disease is present*". Classification rate for the training data is 96.51%. As above, only two rules for each category are presented:

Rule H_4C_ c1_2: **IF** age is of the range <26,55> **OR** <62,84>
AND *resting blood pressure (mm Hg)* is of the range <91,164.5>
AND *serum cholestoral (mg/dl)* is of the range <119,228.5>
OR <248.5,280> **OR** <299.5,638>
AND *maximum heart rate* achieved is of the range <115.5,175.5>
OR <182.5,193>
AND no *major vessels,* **OR** 1 **OR** 2 is colored by *flourosopy*
AND *thal* is *normal*
THEN *heart disease* is *absent (<50% diameter narrowing)*

Confidence level (bbm_T): 0.980

Rule H_4C_ c1_3: **IF** *resting blood pressure (mm Hg)* is of the range <91,129.5>
OR <138.5,150.5> **OR** <164.5,220>
AND *serum cholestoral (mg/ dl)* is of the range <119,172>
OR <186,196.5> **OR** <205.5,228.5> **OR** <234.5,242.5>
OR <256.5,264.5> **OR** <299.5,325.5> **OR** <357,638>
AND *maximum heart rate* achieved is of the range <144.5,193>
AND no *major vessels* **OR** 1 is colored by *flourosopy*
THEN heart disease is absent (<50% diameter narrowing)
Confidence level (bbm_T): 0.971

Rule H_4C_ c2_1: **IF** *age* is of the range <26,68>
AND the type of *Chest Pain* is *asymptomatic*
AND *serum cholestoral (mg/ dl)* is of the range <119,172>
OR <213.5,222.5> **OR** <228.5,242.5> **OR** <248.5,256.5>
OR <264.5,270.5> **OR** <280,299.5> **OR** <325.5,638>
AND *maximum heart rate* achieved is of the range <63,156.5>
OR <169.5,175.5>
THEN heart disease is present (>50% diameter narrowing)
Confidence level (bbm_T): 0.971

Rule H_4C_ c2_5: **IF** *resting blood pressure (mm Hg)* is of the range <91,120.5>
OR <138.5,164.5>
AND *serum cholestoral (mg/ dl)* is of the range <172,186>
OR <228.5,234.5> **OR** <242.5,248.5> **OR** <264.5,270.5>
OR <299.5,308.5>
AND *maximum heart rate* achieved is of the range <123.5,129.5>
OR <144.5,169.5> **OR** <175.5,182.5>
AND *ST depression (induced by exercise relative to rest)* is of the range
<- 1.0,0.65> **OR** <2.15,6.8>
THEN heart disease is present (>50% diameter narrowing)
Confidence level (bbm_T): 0.938

Genetic-based approach has lead to thirty-eight rules: seventeen for "*heart disease is absent*" and twenty-one for "*heart disease is present*". Classification rate, for the training data, is 89.9%. The rules are:

Rule H_GB_ c1_5:**IF** *thal is normal* **OR** has a *fixed defect*
AND age is <36,56>
AND the type of *Chest Pain* is *typical angina* **OR** *atypical angina*

THEN heart disease is absent (<50% diameter narrowing)

Confidence level (bbm_T): 0.960

Rule H_GB_ c1_8:**IF** *thal is normal* **OR** has a *fixed defect*

AND age is <64,82>

AND the type of *Chest Pain* is *non-anginal*

THEN heart disease is absent (<50% diameter narrowing)

Confidence level (bbm_T): 0.889

Rule H_GB_ c2_19:**IF** *thal* has a *reversable defect*

AND age is <56,64>

AND the number of *major vessels* is colored by *flourosopy* is 1 **OR** 3

AND the type of *Chest Pain* is *asymptomatic*

THEN heart disease is present (>50% diameter narrowing)

Confidence level (bbm_T): 0.929

Rule H_GB_ c2_20:**IF** *thal* has a *reversable defect*

AND age is <56,64>

AND the number of *major vessels* is colored by *flourosopy* is 2

THEN heart disease is present (>50% diameter narrowing)

Confidence level (bbm_T): 0.900

5.3.2 Validation of Rules

The results of the validation process are presented in Table 18, Table 19 and Table 20. In the case of the rules generated by C5 method, the best rules of the category "*hear disease is absent*" have kept their highest belief values. In the case of rules H_C5_c1_1 and H_C5_c2_2, their bbm values increased after the validation phase. However, in the category "*heart disease is present*" the belief value of the rule H_C5_c2_2 has been suppress by the belief value of the rule H_C5_c2_3, and this rule has become one of the best rules (there are some cases where bbm values are not changes – it means that a rule has not been satisfied by any validation data point). The new best rule H_C5_c2_3 is presented below.

Table 18.Original and updated values of bbm for rules generated by C5

Rule Number	Original (bba_T)	After Validation (bba_{update})
Category: heart disease is absent		
H_C5_ c1_1	**0.962**	**0.994**
H_C5_ c1_2	**0.935**	**0.978**
H_C5_ c1_3	0.857	0.977
H_C5_ c1_4	0.833	0.909
H_C5_ c1_5	0.800	0.909
H_C5_ c1_6	0.779	0.910
Category: heart disease is present		
H_C5_ c2_1	**0.952**	**0.986**
H_C5_ c2_2	**0.938**	0.938
H_C5_ c2_3	0.846	**0.971**
H_C5_ c2_4	0.842	0.842
H_C5_ c2_5	0.833	0.833
H_C5_ c2_6	0.800	0.800
H_C5_ c2_7	0.747	0.846
H_C5_ c2_8	0.688	0.423

Rule H_C5_ c2_3: **IF** *ST depression* (induced by exercise relative to rest)
is of the range <2.6,6.8)
THEN *heart disease* is *present (>50% diameter narrowing)*
Confidence level (bbm_{UPDATE}): 0.971

The validation process has been also performed for rules generated by 4C, Table 19. In the case of rules H_4C_c1_2, H_4C_c1_3 and H_4C_c2_1 their bbm values have increased and the rules are still among the best ones. In the case of the fourth rule H_4C_c2_5, its bbm value has decreased. At the same time, the bbm value of the rule H_4C_c2_6 has stayed the same and has become higher than the updated bbm value of H_4C_c2_5. As the result a set of best rules has been modified. The new rule is shown below.

Table 19.Original and updated values of bbm for rules generated by 4C

Rule Number	Original (bba_T)	After Validation (bba_{update})
Category: heart disease is absent		
H_4C_ c1_1	0.875	0.955
H_4C_ c1_2	**0.980**	**0.993**
H_4C_ c1_3	**0.971**	**0.989**
H_4C_ c1_4	0.926	0.962
H_4C_ c1_5	0.955	0.984
H_4C_ c1_6	0.889	0.727
Category: heart disease is present		
H_4C_ c2_1	**0.971**	**0.996**
H_4C_ c2_2	0.750	0.600
H_4C_ c2_3	0.800	0.667
H_4C_ c2_4	0.875	0.875
H_4C_ c2_5	**0.938**	0.789
H_4C_ c2_6	0.923	**0.923**
H_4C_ c2_7	0.833	0.909

Rule H_4C_ c2_6: **IF** *resting blood pressure (mm Hg)* is of the range <129.5,138.5>
OR <150.5,220>
AND *serum cholestoral (mg/ dl)* is of the range <172,186>
OR <205.5,213.5> **OR** <225.5,228.5> **OR** <234.5,242.5>
OR <280,289.5> **OR** <308.5,325.5>
AND *maximum heart rate* achieved is of the range <63,106.5>
OR <123.5,138.5> **OR** <144.5,150.5> **OR** <156.5,175.5>
AND the *slope of the peak exercise ST segment* is *flat*
THEN heart disease is present (>50% diameter narrowing)
Confidence level (bbm_{UPDATE}): 0.923

The validation process performed on genetic-based generated rules has brought changes in both categories; see Table 20. The rule H_GB_c1_8 has been replaced by the rule H_GB_c1_2. The bbm value of H_GB_c1_2 has improved a lot. In the category "*heart disease is present*", the rule H_GB_c2_20 has been surpassed by the rule H_GB_c2_14. New rules are shown below.

Table 20.Original and updated values of bbm for rules generated by GB

Rule Number	Original (bba_T)	After Validation (bba_{update})
Category: heart disease is absent		
H_GB_ c1_1	0.750	0.900
H_GB_ c1_2	0.844	**0.980**
H_GB_ c1_3	0.875	0.972
H_GB_ c1_4	0.750	0.750
H_GB_ c1_5	**0.960**	**0.990**
H_GB_ c1_6	0.813	0.765
H_GB_ c1_7	0.571	0.727
H_GB_ c1_8	**0.889**	0.889
H_GB_ c1_9	0.750	0.857
H_GB_ c1_10	0.750	0.750
H_GB_ c1_11	0.667	0.667
H_GB_ c1_12	0.750	0.750
H_GB_ c1_13	0.667	0.667
H_GB_ c1_14	0.667	0.667
H_GB_ c1_15	0.800	0.923
H_GB_ c1_16	0.750	0.750
H_GB_ c1_17	0.625	0.625
Category: heart disease is present		
H_GB_ c2_1	0.667	0.667
H_GB_ c2_2	0.600	0.429
H_GB_ c2_3	0.875	0.933
H_GB_ c2_4	0.667	0.667
H_GB_ c2_5	0.800	0.667
H_GB_ c2_6	0.500	0.500
H_GB_ c2_7	0.800	0.889
H_GB_ c2_8	0.600	0.600
H_GB_ c2_9	0.800	0.889
H_GB_ c2_10	0.500	0.500
H_GB_ c2_11	0.600	0.600
H_GB_ c2_12	0.750	0.750
H_GB_ c2_13	0.750	0.750
H_GB_ c2_14	0.882	**0.968**
H_GB_ c2_15	0.667	0.667
H_GB_ c2_16	0.667	0.667

H_GB_ c2_17	0.833	0.833
H_GB_ c2_18	0.500	0.500
H_GB_ c2_19	**0.929**	**0.951**
H_GB_ c2_20	**0.900**	0.900
H_GB_ c2_21	0.800	0.923

Rule H_GB_ c1_2: **IF** *thal is normal* **OR** has a *fixed defect*
AND age is <36,56>
AND the type of *Chest Pain* is *non-anginal*
THEN heart disease is absent (<50% diameter narrowing)
Confidence level ($bbm_{UPDATED}$): 0.980

Rule H_GB_ c2_14:**IF** *thal* has a *reversable defect*
AND age is <36,56>
AND the type of *Chest Pain* is *asymptomatic*
THEN heart disease is present (>50% diameter narrowing)
Confidence level ($bbm_{UPDATED}$): 0.968

5.3.3 Analysis of IF-THEN Rules

Development and validation stages result in three sets of IF-THEN rules (each set represents a single model) and bbm values associated with them. Let's take a closer look at the ones, which have the highest values of bbm in each set in each category (marked by bold fonts in Table 18, Table 19 and Table 20 in "After Validation" column).

An interesting observation can be made when all these rules, from category "*heart disease is absent*", are compared. Rules H_C5_ c1_1 and H_4C_c1_2 are similar. It looks like all of them use attributes "*age*", "*major vessels are coloured*" and "*thal*" to predict absence of heart disease. The most specific rule is generated by 4C method. It "includes" the rule H_4C_c1_1 and overlaps with the rule H_GB_c1_5. It is the following form:

IF age is of the range <26,55> **OR** <62,84>
AND *resting blood pressure (mm Hg)* is of the range <91,164.5>
AND *serum cholestoral (mg/ dl)* is of the range <119,228.5>
OR <248.5,280> OR <299.5,638>

AND *maximum heart rate* achieved is of the range <115.5,175.5>
OR <182.5,193>
AND no *major vessels*, **OR** 1 **OR** 2 is colored by *flourosopy*
AND *thal* is *normal*
THEN *heart disease* is *absent (<50% diameter narrowing)*

Its high bbm value of 0.993 makes it a good candidate for practical applications in a heart disease classification process.

In the case of the category "*heart disease is present*", the rule H_GB_c2_19 is similar to rules H_C5_c2_1 and H_4C_c2_1. The attributes common across all three rules are "*chest pain*". This indicates importance of this attribute. Other attributes "shared" by the rules H_GB_c2_19 and H_C5_c2_1 is the attribute "*major vessel is colored by fluoroscopy*", and the "*age*" attribute between H_GB_c2_19 and H_4C_c2_1. Other attributes, which seem important, are "*sex*" (the rule H_C5_c2_1), "*serum cholesterol*" and "*max heart rate*" (H_4C_c2_1).

5.3.4 Prediction of Heart Disease

The constructed multi-model estimation system has been used to predict number of components to be examined for all data points from the testing set. Two aspects of the results are important – classification rate and confidence levels assigned to obtained classifications.

The testing set contains 50 data points. Using only a single set of rules generated by C5, 4C and GB the classification rates are 78.0%, 52.0% and 78.0% respectively. Application of the multi-model estimation system increased classification rate to 84.0% (it is translated to forty-two proper classifications, comparing to thirty-nine obtained for C5 and GB rules). Out of all testing data points, sixteen data points satisfy rules, which indicate contradicting classification. The rest of data points, thirty-four in total, have satisfied rules that uniquely identify point's belonging to a single category (most data points have satisfied at least a single rule from each model, however some points have been satisfied only rules coming from two models). In all these cases probability BetP that a given data point belongs to a single category is very high 0.99. More interesting is the case of sixteen data points that have lead to non-unanimous classification results. From all these data

points three are discussed below. The rules that are satisfied by these three points are presented in Table 21.

Table 21.Rules satisfied by the three "confusing" data points

	C5		4C		GB		Original Category	Predicted Category
	I	II	I	II	I	II		
1		c2_7			c1_14		I	II
2		c2_7			c1_16		II	II
3	c1_1 c1_6			c2_1		c2_2	I	I

It can be easily observed that each of these points is not classified to a single category. Rules generated by C5 have a problem with the data point no. 1, rules generated by 4C are misleading for points no. 2 (there is no satisfied rules for points 2 and 3), in the case of genetically generated rules points number 2 and 3 are wrongly classified. Application of TBM has given a set of BetP values. Table 22 shows the value of probabilities of belonging of each of these three data points to a single category.

Table 22.BetP values for the three "confusing" data points

	Data Points		
	1	2	3
BetP (Category I)	0.295	0.369	0.824
BetP (Category II)	0.705	0.631	0.176

PART III: SOFTWARE MAINTENANCE

6 Software Maintenance Data Set

The software maintenance data used for this thesis is from the Architecture Research Facility (ARF) Error Dataset: Data collected and analyzed by David Weiss on the development of the Architecture Research Facility (ARF) at the Naval Research Laboratories (NRL). ARF is one of NASA's projects. This project was to aid the rapid simulation of different computer architectures for research and evaluation purposes.

The original dataset consists of 117 error reports dealing with 143 errors that were isolated and corrected during the ARF development. The dataset also includes 253 records describing each component composing the project. It also includes records describing each routine composing the project and a file containing project description data that contains two sets of data – Error report data, Module descriptive data and System descriptive data

In the error report data, one record is provided for each error report data and contains the dates the errors was observed and corrected, type of error, the routine in which the error was isolated etc. Also, in the module descriptive data, one record is available for each routine in the system containing the size of the routine, the number of preprocessor statements, the function of the module or routine etc

A free format file is also provided showing the system descriptive data that contains the size and other attributes of the project, a description of the development environment and a description of the development methodologies used on the project. The system was developed using FORTRAN and contains 21,831 Source lines of code written in 44 man months (10 months). Formal code reviews and information hiding were used but it lacked a chief programmer team and librarian.

6.1 Data Selection and Extraction

The original data was in a text format containing two files - Module descriptive data schema and the Error report schema. It was then made in excel readable format that makes the different error reports and module description to be separated column-wise thus making each data description distinct. A new file is then generated by combining the Module descriptive data schema and the

Error report schema. For each component code of the module descriptive dataset, its relation in the error report dataset is searched for and then merged with the module descriptive data. The new file contains 323 data points containing all the attributes of the module descriptive data set and the error report data set. The data is generally of good quality, and very few data items are missing.

6.2 Data Pre-processing

From the new data set, which has 323 points, 3 new data sets that concern software maintenance where then extracted. The first is related to **time for isolation of defects** and has 129 data points with 8 attributes, the second set is concerned with **the number of component examined** and has 129 data points with 8 attributes and the third is related with **effort for elimination of defects** containing 129 data points and has 9 attributes. The three sets of data their attributes are shown in Table 23, Table 24 and Table 25 below.

Table 23.Data attributes to estimate Time needed for Isolation of Defect

Attribute Name	Attribute Type	Attribute Range
INPUT		
Lines of Code	continuous	<3, 635>
Number of Comments	continuous	<0, 360>
Number of Preprocessor Statements	continuous	<0, 42>
Subjective Complexity	discrete	Easy, Moderate, Hard
Functionality	discrete	Computational, Control Data_Accessing, Error_Handling
Type of Defect	discrete	Requirements_Incorrect Functional_Spec._Incorrect Single_Design_Error, Other_Error Multiple_Design_Error Language_Use_Error Clerical_Error, Multiple_Error
OUTPUT		
Time to Isolate Error	discrete	Less than 1 Hour, More than 1 Hour

Table 24.Data attributes to estimate Number of Components Examined

Attribute Name	Attribute Type	Attribute Range
INPUT		
Type of Defect	discrete	Requirements_Incorrect Functional_Spec._Incorrect Single_Design_Error, Other_Error Multiple_Design_Error Language_Use_Error Clerical_Error, Multiple_Error
Phase When Defect Entered System	discrete	Requirements_Definition, Design Functional_Specification, Code_Testing
Lines of Code	continuous	<3 , 635>
Number of Comments	continuous	<0, 360>
Number of Preprocessor	continuous	<0, 42>
Subjective Complexity	discrete	Easy, Moderate, Hard
Functionality	discrete	Computational, Control Data_Accessing, Error_Handling
OUTPUT		
Number of Components Examined	discrete	Single_Component, Multiple_Components

Table 25.Data attributes to estimate Effort needed for Elimination of Defect

Attribute Name	Attribute Type	Attribute Range
INPUT		
Type of Defect	discrete	Requirements_Incorrect Functional_Spec._Incorrect Single_Design_Error, Other_Error Multiple_Design_Error Language_Use_Error Clerical_Error, Multiple_Error
Phase When Defect Entered System	discrete	Requirements_Definition, Design Functional_Specification, Code_Testing
Number of Examined Components	discrete	Single_Component, Multiple_Components
Lines of Code	continuous	<3 , 635>
Number of Comments	continuous	<0, 360>
Number of Pre-processor Statements	continuous	<0, 42>
Subjective Complexity	discrete	Easy, Moderate, Hard
Functionality	discrete	Computational, Control Data_Accessing, Error_Handling
OUTPUT		
Effort needed for Elimination of Defect	discrete	Less than 1 hour More than 1 hour but less than 1 day More than 1 day

The data points have been divided into three sets: training set of 86 data points, validation set containing 20 data points, and testing set which has 23 data points.

7 Isolation of Software Defects

A very important management task related to corrective maintenance processes is to estimate time needed for isolation of a single defect. It can be said that this estimation has a large impact on a possible outcome of a planning procedure. Planning process should be improved when a better understanding of relationships is gained and confidence levels of predicted values are known.

7.1 Model Building

Three different, independent methods have been used to build three independent models. All three methods are built based on the same training data. Extracted rules try to describe the following relation:

Time.to.Isolate = f (no of statements,
no of comments,
no of pre-processor statements,
complexity,
functionality,
type of defect)

Three sets of **IF-THEN** rules have been created. The first set is obtained using C5 tool, the second using 4C tool, and the third using the evolutionary-based method. Each of the used techniques processed data differently, what is reflected in quite different rules that are extracted.

C5 generated eleven rules. Five rules for "time to isolate less than 1 hour" and six rules for "time to isolate more than 1 hour". A model generated by C5 gives classification of 80.2% for training data. Due to space constrains only two the most important rules generated for each output category are presented below. For each rule, a confidence level (bbm) calculated based on training data is shown.

Rule C5_c1_1: **IF** defect type is *Clerical_Error*
THEN time to isolate is less than 1 hour
Confidence level (bbm_T): 0.938

Rule C5_c1_1: **IF** defect type is *Functional_Spec._Incorrect*
THEN time to isolate is less than 1 hour
Confidence level (bbm_T): 0.846
Rule C5_c2_1: **IF** lines of code is less than 207

AND complexity is *Hard*
THEN time to isolate is more than 1 hour
Confidence level (bbm_T): 0.833

Rule C5_c2_1: **IF** defect type is *Other*
THEN time to isolate is more than 1 hour
Confidence level (bbm_T): 0.800

4C has generated seven rules for "time to isolate less than 1 hour", and five for "time to isolate more than 1 hour". Classification rate is 79.4% for the training data. As above, only two rules for each case are presented:

Rule 4C_c1_1: **IF** defect type is *Functional_Spec._Incorrect* **OR** *Clerical_Error*
OR *Single_Design_Error* **OR** *Requirements_Incorrect*
AND number of comments is larger than 36
THEN time to isolate is less than 1 hour
Confidence level (bbm_T): 0.897

Rule 4C_c1_3: **IF** defect type is *Clerical_Error* **OR** *Language_Use_Error*
OR *Functional_Spec._Incorrect* **OR** *Requirements_Incorrect*
AND lines of code is more than 80
AND complexity is *Moderate* **OR** *Hard*
THEN time to isolate is less than 1 hour
Confidence level (bbm_T): 0.909

Rule 4C_c2_2: **IF** defect type is *Language_Use_Error* **OR** *Multiple_Design_Error* **OR** *Other*
AND lines of code is in more than 80 but less than 130
AND number of comments is larger than 34 but less than 62
AND number of preprocessor statements is larger than 51
AND functionality is *Computational* **OR** *Error_Handling* **OR** *Data_Accessing*
THEN time to isolate is more than 1 hour
Confidence level (bbm_T): 0.800

Rule 4C_c2_4: **IF** defect type is *Language_Use_Error* **OR** *Multiple_Design_Error* **OR** *Other*
AND number of comments less than 34 **OR** more than 111
AND functionality is *Computational* **OR** *Error_Handling*
THEN time to isolate is more than 1 hour
Confidence level (bbm_T): 0.857

Genetic-based approach has lead to nineteen rules: eleven for "time to isolate less than 1 hour" and eight for "time to isolate more than 1 hour". Classification rate is 84.9% for the training data. The rules are:

Rule GB_c1_1: **IF** defect type is *Clerical_Error*
THEN time to isolate is less than 1 hour
Confidence level (bbm_T): 0.938

Rule GB_c1_10: **IF** defect type is *Functional_Spec._Incorrect*
THEN time to isolate is less than 1 hour
Confidence level (bbm_T): 0.846

Rule GB_c2_2: **IF** defect type is *Multiple_Error*
AND number of comments is less than 52
THEN time to isolate is more than 1 hour
Confidence level (bbm_T): 0.833

Rule GB_c2_7: **IF** defect type is *Language_Use_Error*
AND number of comments more than 82 **OR** less than 132
AND complexity is *Easy*
THEN time to isolate is more than 1 hour
Confidence level (bbm_T): 0.800

7.2 Validation of Rules

An important concept included in the proposed approach is related to validation of the original set of rules. In order to do this the second set of data, called a validation set, is used. All the previously generated rules are checked against data points from the validation set. A new belief value bbm_V is calculated for each rule, and is used to update bbm_T. As the result updated belief values are created. The effect of this process is presented in Table 26.

Table 26.Original and updated values of bbm for rules generated by C5

Rule Number	Original (bba_T)	After Validation (bba_{update})
Category I: less than 1 hour		

C5_ c1_1	**0.938**	**0.957**
C5_ c1_2	**0.846**	**0.892**
C5_ c1_3	0.750	0.750
C5_ c1_4	0.727	0.842
C5_c1_5	0.559	0.792
Category II: more than 1 hour		
C5_ c2_1	**0.833**	**0.833**
C5_ c2_2	**0.800**	**0.800**
C5_ c2_3	0.667	0.667
C5_ c2_4	0.643	0.474
C5_ c2_5	0.636	0.724
C5_ c2_6	0.600	0.429

The best rules originally generated by C5 have kept their high values. In the case of rules C5_c1_1 and C5_c2_2, their bbm values increased after the validation phase. Two of the updated bbm values are smaller that the original ones (there are some cases where bbm values are not changes – it means that a rule has not been satisfied by any validation data point).

The same process has been performed for rules generated by 4C, Table 27. In the case of a rule 4C_c1_1 increased its bbm value and the rules is still among the best ones. The other two rules: 4C_c2_2 and 4C_c2_4 have the same values of bbm. However, the bbm value of the rule 4C_c1_3 has decreased and is smaller. In such case bbm value of the rule 4C_c1_7 is larger. As the result a set of best rules has been modified. The new rule is shown below.

Table 27.Original and updated values of bbm for rules generated by 4C

Rule Number	Original	After Validation
Category I: less than 1 hour		
4C_ c1_1	**0.897**	**0.972**
4C_ c1_2	0.656	0.851
4C_ c1_3	**0.909**	0.857
4C_ c1_4	0.524	0.767
4C_ c1_5	0.857	0.857

4C_ c1_6	0.833	0.833
4C_ c1_7	0.889	**0.889**
Category II: more than 1 hour		
4C_ c2_1	0.643	0.545
4C_ c2_2	**0.800**	**0.800**
4C_ c2_3	0.667	0.667
4C_ c2_4	**0.857**	**0.857**
4C_ c2_5	0.429	0.273

Rule 4C_c1_7: **IF** number of comments in less than 36
AND functionality is *Control*
THEN time to isolate is more than 1 hour
Confidence level (bbm_{UPDATE}): 0.889

The validation process performed on genetic-based generated rules has not brought any changes; see Table 28.

Table 28.Original and updated values of bbm for rules generated by GB

Rule Number	Original	After Validation
Category I: less than 1 hour		
GB_ c1_1	**0.938**	**0.957**
GB_ c1_2	0.667	0.667
GB_ c1_3	0.667	0.667
GB_ c1_4	0.750	0.857
GB_ c1_5	0.667	0.667
GB_ c1_6	0.667	0.667
GB_ c1_7	0.600	0.750
GB_ c1_8	0.667	0.800
GB_ c1_9	0.667	0.667
GB_ c1_10	**0.846**	**0.892**
GB_ c1_11	0.750	0.750
Category II: more than 1 hour		
GB_ c2_1	0.778	0.778
GB_ c2_2	**0.833**	**0.833**

GB_ c2_3	0.600	0.600
GB_ c2_4	0.750	0.750
GB_ c2_5	0.571	0.400
GB_ c2_6	0.556	0.556
GB_ c2_7	**0.800**	**0.889**
GB_ c2_8	0.800	0.800

7.3 Analysis of IF-THEN Rules

Development and validation stages result in three sets of **IF-THEN** rules (each set represents a single model) and bbm values associated with them. Let's take a closer look at the ones, which have the highest values of bbm in each set in each category (marked by bold fonts in (Table 26, Table 27 and Table 28) in "After Validation" column). For category "time to isolate less than 1 hour" the range of bbm values is from 0.892 to 0.972. The rules with highest bbm values are 4C_ c1_1 with bbm of 0.972, and C5_ c1_1 and GB_c1_1 with bbm of 0.957.

An interesting observation can be made when all these rules, from category "time to isolate less than 1 hour". There are two pairs of identical rules generated by C5 and genetic approach. One of the pair is C5_c1_1 and GB_c1_1 with bbm of 0.957, and the other C5_c1_2 and GB_c1_10 with bbm of 0.892. Such situation and high bbm values of these rules indicate that someone can use both of them to support prediction for time needed to isolate defects. The conclusion which can be derived here is: if 'defect type' is Clerical_*Error* **OR** *Functional_Sepecification_Incorrect* then time to isolate this kind of defect is less than one hour.

Rule C5_c1_1: **IF** defect type is *Clerical_Error*
THEN time to isolate is less than 1 hour

Rule GB_c1_1: **IF** defect type is *Clerical_Error*
THEN time to isolate is less than 1 hour

Rule C5_c1_2: **IF** defect type is *Functional_Spec._Incorrect*
THEN time to isolate is less than 1 hour

Rule GB_c1_10: **IF** defect type is *Functional_Spec._Incorrect*
THEN time to isolate is less than 1 hour

In the case of the category "time to isolate more than 1 hour", the rules have bbm value in a smaller range: from 0.800 up to 0.889. This indicates that it is difficult to find good rules describing a relationship among attributes of software components and a defect isolation time. The best rule in this case is:

Rule GB_c2_7: **IF** defect type is *Language_Use_Error*
AND number of comments more than 82 **OR** less than 132
AND complexity is *Easy*
THEN time to isolate is more than 1 hour

7.4 Prediction of Time for Isolation of Defects

The constructed multi-model estimation system has been used to predict time to isolate defects for all data points from the testing set. Two aspects of the results are important – classification rate and confidence levels assigned to obtained classifications.

The testing set contains 23 data points. Using only a single set of rules generated by C5, 4C and GB the classification rates are 65.22%, 47.83% and 60.87% respectively. Application of the multi-model estimation system increased classification rate to 69.57%. Out of all testing data points, eight data points satisfy rules, which indicate contradicting classification. The rest of data points has satisfied rules that uniquely identify point's belonging to a single category (most data points have satisfied at least a single rule from each model, some points have satisfied only rules coming from two models). In all these cases probability BetP that a given data point belongs to a single category is very high 0.999. More interesting is the case of eight data points that have lead to non-unanimous classification results. All rules that are satisfied by these eight points are presented in Table 29.

Rules generated by C5 have problems with data points no. 1, 4, 6, 7 and 8, rules generated by 4C are misleading for all eight points, in the case of genetically generated rules points no. 1, 2, 3, 4, 6 and 8 are wrongly classified. Application of TBM has given a set of BetP values. Table 30shows the value of probabilities of belonging of each of the eight data points to a single category.

A data point no. 8 represents a very interesting case. Only a single rule generated by C5 indicates membership of this point to the category I. Two other rules, one generated by C5 and other generated by genetic-based approach, indicate that the point belongs to the category II. However, a

higher bbm value of the C5 rule (comparing to bba's of the other C5 rule and the GB rule) has changed a verdict of the System. It can be seen that probability of the point belonging to the category I is 0.503, and to the category II is 0.497. Such values of probabilities indicate that the point "is on the boarder" between two categories.

Table 29.Rules satisfied by the eight "confusing" data points

	C5		4C		GB		Original Category	Predicted Category
	I	II	I	II	I	II		
1	c1_2 c1_5		c1_1 c1_2 c1_4	c2_5	c1_10		II	I
2		c2_4			c1_4		II	I
3		c2_5			c1_6		II	II
4		c2_1	c1_2	c2_1		c2_8	I	II
5	c1_2 c1_5		c1_1 c1_2 c1_4	c2_5	c1_10		I	I
6	c1_5		c1_2 c1_4	c2_1	c1_7		II	I
7	c1_4	c2_3				c2_1	II	II
8	c1_4	c2_4				c2_3	I	I

Table 30.BetP values for the eight "confusing" data points

	Data Points							
	1	2	3	4	5	6	7	8
BetP (Category I)	0.999	0.802	0.474	0.089	0.999	0.997	0.240	0.503
BetP (Category II)	0.001	0.198	0.526	0.911	0.001	0.003	0.760	0.497

8 Number of Components Examined

Depending on the location of the defect, i.e. how complex and important a software component that contains the defect is, a different number of components have to be checked to ensure that impact of the defect is well understood and the defect can be removed. Questions, which can be asked here are: what kinds of relationships exist between number of components that have to be examined during elimination of defects and attributes of a software component that contains a defect? How these relationships can be formulated in the context of different defects and time when they enter the system? Another important element is a confidence level which can be assigned to found relationships. To determine these aspects, an approach proposed in the paper is applied.

8.1 Model Building

Building **IF-THEN** based models is the first step. Models are built based on data points from a training set (with attributes presented in Table 24). They are representations of the following relation:

Number of Components Examined = relation (Type of Defect,
Phase When Defect Entered System,
No of Statements,
No of Comments,
No of Pre-processor Statements,
Complexity,
Functionality)

Three different, independent methods have been used to build three independent models. All three methods are built based on the same training data therefore all extracted rules try to describe the same data points.

The method that has been used as the first is C5. As the result eleven rules have been generated: four for "single component examined", and seven for "multiple components examined". A model generated by C5 gives 88.4% correct classifications for the training data. Due to space constrains

only two the most important rules generated for each output category – single component, multiple components – are presented below. For each rule, a confidence level (bbm) is shown together with a total number of data points satisfying all conditions in the antecedent, and a number of data points that satisfy the antecedent and belong to the category indicated by the consequent.

Rule NC_C5_c1_1: **IF** complexity is *Easy*
THEN single component examined

Confidence level (bbm_T):	0.791
Data points satisfying the antecedent:	41
Data points satisfying the antecedent and with matching consequent:	33

Rule NC_C5_c1_2: **IF** phase when defect entered system is *Design*
AND complexity is *Hard*
THEN single component examined

Confidence level (bbm_T):	0.750
Data points satisfying the antecedent:	2
Data points satisfying the antecedent and with matching consequent:	2

Rule NC_C5_c2_1: **IF** defect type is *Language_Use_Error*
AND lines of code is less than **OR** equal 158
AND number of comments is more than 102
THEN multiple components examined

Confidence level (bbm_T):	0.833
Data points satisfying the antecedent:	4
Data points satisfying the antecedent and with matching consequent:	4

Rule NC_C5_c2_2: **IF** defect type is *Functional_Spec._Incorrect*
AND lines of code is more than 134
AND complexity is *Easy*
THEN multiple components examined

Confidence level (bbm_T):	0.800
Data points satisfying the antecedent:	3
Data points satisfying the antecedent and with matching consequent:	3

4C, on the other hand, has generated six rules for "single component examined", and five for "multiple components examined". Classification rate for the training data is 96.51%. As above, only two rules for each category are presented:

Rule NC_4C_c1_1: **IF** defect type is *Single_Design_Error* **OR** *Multiple_Design_Error*
OR *Language_Use_Error* **OR** *Functional_Spec._Incorrect* **OR** *Multiple_Error*
AND lines of code is in the range <58, 135>
AND number of preprocessors statements is less than 9
AND complexity is *Easy* **OR** *Moderate*
AND functionality is *Computational* **OR** *Data_Accessing*
OR *Error_Handling*
THEN single component examined

Confidence level (bbm_T):	0.950
Data points satisfying the antecedent:	18
Data points satisfying the antecedent and with matching consequent:	18

Rule NC_4C_c1_5: **IF** defect type is *Multiple_Design_Error* **OR** *Clerical_Error*
OR *Functional_Spec._Incorrect*
AND functionality is *Error_Handling or Control* **OR** *Data_Accessing*
THEN single component examined

Confidence level (bbm_T):	0.941
Data points satisfying the antecedent:	15
Data points satisfying the antecedent and with matching consequent:	15

Rule NC_4C_c2_2: **IF** defect type is *Single_Design_Error* **OR** *Requirements_Incorrect*
OR *Functional_Spec._Incorrect* **OR** *Multiple_Error* **OR** *Other_Error*
AND lines of code is in the ranges <85, 108> **OR** <168, 207>
AND number of comments in the range <45, 128>
AND number of pre-processors statements is more than 3
AND complexity is *Easy* **OR** *Moderate*
THEN multiple components examined

Confidence level (bbm_T):	0.889
Data points satisfying the antecedent:	7

Data points satisfying the antecedent and with matching consequent: 7

Rule NC_4C_c2_3: **IF** defect type is *Requirements_Incorrect* **OR** *Multiple_Error*
OR *Other_Error*
AND lines of code is less than 84
AND complexity is *Moderate*
THEN multiple components examined

Confidence level (bbm_T): 0.857

Data points satisfying the antecedent: 5

Data points satisfying the antecedent and with matching consequent: 5

Genetic-based approach has lead to twenty-one rules: fourteen for "single component examined " and seven for "multiple components examined". Classification rate, for the training data, is 93.0%. The rules are:

Rule NC_GB_c1_2: **IF** defect type is *Single_Design_Error* **OR** Multiple_Design_Error
OR Multiple_Error
AND lines of code is less than 141
AND complexity is *Easy*
THEN single component examined

Confidence level (bbm_T): 0.909

Data points satisfying the antecedent: 9

Data points satisfying the antecedent and with matching consequent: 9

Rule NC_GB_c1_9: **IF** defect type is *Clerical_Error* **OR** *Requirements_Incorrect*
AND functionality is *Control* **OR** *Error_Handling*
THEN single component examined

Confidence level (bbm_T): 0.889

Data points satisfying the antecedent: 7

Data points satisfying the antecedent and with matching consequent: 7

Rule NC_ GB_c2_2: **IF** defect type is *Functional_Spec._Incorrect*
AND lines of code is in the range <141, 481>
AND complexity is *Easy*

AND functionality is *Computational*

THEN multiple components examined

Confidence level (bbm_T):	0.800
Data points satisfying the antecedent:	3
Data points satisfying the antecedent and with matching consequent:	3

Rule NC_GB_c2_3:**IF** defect type is *Single_Design_Error* **OR** *Multiple_Design_Error*

OR *Multiple_Error*

AND lines of code is less than 141

AND complexity is *Moderate* **OR** *Hard*

THEN multiple components examined

Confidence level (bbm_T):	0.857
Data points satisfying the antecedent:	5
Data points satisfying the antecedent and with matching consequent:	5

8.2 Validation of Rules

An important concept included in the proposed approach is related to validation of the original set of rules. In order to do this the second set of data, called a validation set, is used. All the previously generated rules are checked against data points from the validation set. Based on number of $True_Positive_V$ and $False_Positive_V$ data points a new belief value bbm_V is calculated for each rule. These values are used to update bbm_T values. As the result updated belief values are created. The effect of this process is presented in Table 9. The best rules originally generated by C5 have kept their high values. In the case of rules NC_C5_c1_1 and NC_C5_c2_2, their bbm values increased after the validation phase. As it can be seen on the basis of other rules, none of the updated bbm values is smaller that the original one (there are some cases where bbm values are not changes – it means that a rule has not been satisfied by any validation data point).

Table 31.Original and updated values of bbm for rules generated by C5

Rule Number	Original (bba_T)	After Validation (bba_{update})
Category: single component		
NC_C5_ c1_1	**0.791**	**0.883**
NC_C5_ c1_2	**0.750**	**0.750**
NC_C5_ c1_3	0.667	0.667

NC_C5_ c1_4	0.500	0.545
Category: multiple components		
NC_C5_ c2_1	**0.833**	**0.833**
NC_C5_ c2_2	**0.800**	**0.889**
NC_C5_ c2_3	0.800	0.800
NC_C5_ c2_4	0.750	0.750
NC_C5_ c2_5	0.750	0.750
NC_C5_ c2_6	0.667	0.800
NC_C5_ c2_7	0.667	0.800

The same process has been performed for rules generated by 4C, Table 32. In the case of two rules NC_4C_c1_5 and NC_4C_c2_3 their bbm values have increased and the rules are still among the best ones. The other two rules: NC_4C_c1_1 and NC_4C_c2_2 have the same values of bbm. However, the bbm values of other two rules NC_4C_c1_2 and NC_4C_c2_1 have increased and have become higher than bbm values of NC_4C_c1_1 and NC_4C_c2_2. As the result a set of best rules has been modified. The two new rules are shown below.

Table 32.Original and updated values of bbm for rules generated by 4C

Rule Number	Original	After Validation
Category: single component		
NC_4C_ c1_1	**0.950**	0.950
NC_4C_ c1_2	0.929	**0.985**
NC_4C_ c1_3	0.633	0.776
NC_4C_ c1_4	0.889	0.952
NC_4C_ c1_5	**0.941**	**0.988**
NC_4C_ c1_6	0.455	0.455
Category: multiple components		
NC_4C_ c2_1	0.818	**0.900**
NC_4C_ c2_2	**0.889**	0.889
NC_4C_ c2_3	**0.857**	**0.923**
NC_4C_ c2_4	0.667	0.800
NC_4C_ c2_5	0.750	0.750

Rule NC_4C_c1_2: **IF** defect type is *Single_Design_Error* **OR** *Multiple_Design_Error*
OR *Clerical_Error* **OR** *Language_Use_Error* **OR** *Functional_Spec._Incorrect*
OR *Multiple_Error*
AND phase when error entered system is *Requirements_Definition*
OR *Functional_Specification* **OR** *Code_Testing*
AND lines of code is less than 85 **OR** in the range <108, 167>
OR more than207
AND number of comments is less than 29
OR in the range <45, 103> **OR** more than 128
AND number of preprocessors statements is less than 9
THEN single component examined

Confidence level (bbm_{UPDATE}): 0.985

Rule NC_4C_c2_1: **IF** defect type is *Clerical_Error* **OR** *Language_Use_Error*
OR *Requirements_Incorrect* **OR** *Other_Error*
AND phase when error entered system is *Code_Testing*
AND lines of code is less than 54 **OR** more than 135
AND number of comments in the range <29, 45> **OR** more than 62
AND number of preprocessors statements is less than 9
AND functionality is *Computational* **OR** *Control*
THEN multiple components examined

Confidence level (bbm_{UPDATE}): 0.900

The validation process performed on genetic-based generated rules has brought a change in the second category of multiple components; see Table 33. The rule NC_GB_c2_3 has been replaced by NC_GB_c2_7 which bbm value has improved a lot. The bba values of rules from the first category "single component" have improved and stayed the highest. A new rule is the second category is shown below.

Table 33.Original and updated values of bbm for rules generated by GB

Rule Number	Original	After Validation
Category: single component		

NC_GB_ c1_1	0.600	0.692
NC_GB_ c1_2	**0.909**	**0.952**
NC_GB_ c1_3	0.800	0.800
NC_GB_ c1_4	0.750	0.857
NC_GB_ c1_5	0.667	0.667
NC_GB_ c1_6	0.889	0.889
NC_GB_ c1_7	0.750	0.857
NC_GB_ c1_8	0.667	0.667
NC_GB_ c1_9	**0.889**	**0.941**
NC_GB_ c1_10	0.667	0.667
NC_GB_ c1_11	0.800	0.889
NC_GB_ c1_12	0.833	0.833
NC_GB_ c1_13	0.800	0.667
NC_GB_ c1_14	0.667	0.667
Category: multiple components		
NC_GB_ c2_1	0.800	0.800
NC_GB_ c2_2	**0.800**	**0.889**
NC_GB_ c2_3	**0.857**	0.857
NC_GB_ c2_4	0.667	0.667
NC_GB_ c2_5	0.667	0.667
NC_GB_ c2_6	0.800	0.800
NC_GB_ c2_7	0.765	**0.907**

Rule NC_ GB_c2_7: **IF** defect type is *Language_Use_Error*
AND lines of code is in the range <141, 481>
AND complexity is *Moderate* **OR** *Hard*
AND functionality is *Computational*
THEN multiple components examined

Confidence level (bbm_{UPDATE}): 0.907

8.3 Analysis of IF-THEN Rules

Development and validation stages result in three sets of **IF-THEN** rules (each set represents a single model) and bbm values associated with them. Let's take a closer look at the ones, which have the highest values of bbm in each set in each category (marked by bold fonts in Table 31, Table 32

and Table 33 in "After Validation" column). For category "single component examined" the range of bbm values is from 0.750 to 0.988. The rules with highest bbm values are generated by 4C: NC_4C_ c1_2 with bbm of 0.985, and NC_4C_ c1_5 with bbm of 0.988. The rules NC_C5_ c1_1 and NC_C5_ c1_2, generated by C5, have the lowest bbm values among three sets.

An interesting observation can be made when all these rules, from category "single component examined", are compared. Rules NC_GB_ c1_9 and NC_4C_ c1_5 are similar. It looks like both of them "use" attributes 'defect type' and 'functionality' to predict a single component examination. The set of values of the attribute 'defect type' from both rules are overlapping. In the case of attribute 'functionality' values from genetically generated rule are subset of values from 4C-generated rule. Such similarity and high bbm values of these rules indicate that someone can use both of them to support prediction for a single component examination. The conclusion which can be derived here is: if 'defect type' is Clerical_*Error* or *Requirements_Incorrect* and 'functionality' of components with these defects is *Control* or *Error_Handling* then there is a need to examine just one component to be able to understand an impact of these defects and remove them.

Rule NC_4C_c1_5: **IF** defect type is *Multiple_Design_Error* **OR** *Clerical_Error*
OR *Functional_Spec._Incorrect*
AND functionality is *Error_Handling or Control* **OR** *Data_Accessing*
THEN single component examined

Rule NC_GB_c1_9: **IF** defect type is *Clerical_Error* **OR** *Requirements_Incorrect*
AND functionality is *Control* **OR** *Error_Handling*
THEN single component examined

Another pair of rules that draws attention is composed from NC_4C_c1_2 and NC_GB_c1_2. The GB-generated rule shares a lot of common values in the case of attribute 'defect type' with the rule NC_4C_c1_2. Also in the case of the attribute 'lines of code' there is some substantial overlapping. In overall 4C-generated rule seems more specific. Once again high values of bbm make these rules good candidates for prediction activities.

Rule NC_4C_c1_2: **IF** defect type is *Single_Design_Error* **OR** *Multiple_Design_Error*
OR *Clerical_Error* **OR** *Language_Use_Error* **OR** *Functional_Spec._Incorrect*
OR *Multiple_Error*
AND phase when error entered system is *Requirements_Definition*
OR *Functional_Specification* **OR** *Code_Testing*
AND lines of code is less than 85 **OR** in the range <108, 167>
OR more than 207
AND number of comments is less than 29 **OR** in the range <45, 103>
OR more than 128
AND number of preprocessors statements is less than 9
THEN single component examined

Rule NC_GB_c1_2: **IF** defect type is *Single_Design_Error* **OR** Multiple_Design_Error
OR Multiple_Error
AND lines of code is less than 141
AND complexity is *Easy*
THEN single component examined

From C5-generated rules only one rule NC_C5_c1_1 is attractive. It is a very simple rule with relatively high value of bbm, and, what needs more attention for future modifications of the proposed approach, a high coverage – more than 30 data points satisfied both the antecedent and the consequent of this rule. It is also a very intuitive rule. Definitely it is worth consideration during a prediction process.

Rule NC_C5_c1_1: **IF** complexity is *Easy*
THEN single component examined

In the case of the category "multiple components examined", the rules have bbm value in a smaller range: from 0.833 up to 0.923. This indicates that it is difficult to find a dominating rule(s) describing a relationship among attributes of software components and a need for multiple components examination. Inspection of the rules points only to a single pair: 4C-generated rule NC_C5_c2_2 and GB-generated rule NC_ GB_c2_2. There is a perfect match in the case of

attributes 'defect type' and 'complexity'. There is also some overlap existing for the attribute 'lines of codes'. Based on that and high values of bbm, this pair can be treated as a plausible rule describing a relationship among software attributes and a number of components which should be examined to fully understand an impact of a defect *Functional_Spec._Incorrect* and remove it.

Rule NC_C5_c2_2: **IF** defect type is *Functional_Spec._Incorrect*
AND lines of code is more than 134
AND complexity is *Easy*
THEN multiple components examined

Rule NC_ GB_c2_2: **IF** defect type is *Functional_Spec._Incorrect*
AND lines of code is in the range <141, 481>
AND complexity is *Easy*
AND functionality is *Computational*
THEN multiple components examined

8.4 Prediction of Number of Components to be Examined

The constructed multi-model estimation system has been used to predict number of components to be examined for all data points from the testing set. Two aspects of the results are important – classification rate and confidence levels assigned to obtained classifications.

The testing set contains 23 data points. Using only a single set of rules generated by C5, 4C and GB the classification rates are 65.22%, 60.87% and 56.52% respectively. Application of the multi-model estimation system increased classification rate to 73.91% (it is translated to seventeen proper classifications, comparing to fifteen obtained for C5 rules). Out of all testing data points, seven data points satisfy rules, which indicate contradicting classification. The rest of data points, sixteen in total, have satisfied rules that uniquely identify point's belonging to a single category (most data points have satisfied at least a single rule from each model, however some points have been satisfied only rules coming from two models). In all these cases probability BetP that a given data point belongs to a single category is very high 0.99. More interesting is the case of seven data points that have lead to non-unanimous classification results. All rules that are satisfied by these seven points are presented in Table 34.

Table 34.Rules satisfied by the seven "confusing" data points.

	C5		4C		GB		Original Category	Predicted Category
	I	II	I	II	I	II		
1	c1_4	c2_4	c1_3		c1_10		I	I
2	c1_4		c1_5	c2_2	c1_4		I	I
3	c1_4					c2_3	I	II
4	c1_4			c2_1	c1_13		II	II
5	c1_1			c2_1	c1_12		I	I
6		c2_7	c1_2	c2_1		c2_7	II	II
7	c1_4	c2_5		c2_3		c2_3	II	II

It can be easily observed that each of these points is not univocally classified to a single category. Rules generated by C5 have problems with data points number 1, 4 and 7, rules generated by 4C are misleading for points number 2, 3, 5 and 6, in the case of genetically generated rules points number 3 and 4 are wrongly classified. Application of TBM has given a set of BetP values. Table 35 shows the value of probabilities of belonging of each of the seven data points to a single category.

Table 35.BetP values for the seven "confusing" data points

	Data Points						
	1	2	3	4	5	6	7
BetP (Category I)	0.892	0.993	0.207	0.391	0.843	0.014	0.005
BetP (Category II)	0.108	0.007	0.793	0.609	0.157	0.986	0.995

A special attention should be paid to a data point number 4. Rules generated by C5 (NC_C5_c1_4 with bbm of 0.545) and GB (NC_GB_c1_13 with bbm of 0.667) indicate that it should be classified as category I. However, the rule NC_C4_c2_1, which is one of the best C4 rules and has bbm value of 0.900, has a strong influence on the overall result.

9 Effort for Elimination of Software Defects

A very important management task related to corrective maintenance processes is to estimate time needed for elimination of a single defect. It can be said that this estimation has a large impact on a possible outcome of a planning procedure. Planning process should be improved when a better understanding of relationships is gained and confidence levels of predicted values are known. A multi-model estimation system has been constructed for the effort needed for elimination of defects.

9.1 Model Building

Three models have been built using attributes presented in Table 23. They represent the following relation:

Effort for Elimination of Defect = relation (Type of Defect,
Phase When Defect Entered System,
Number of Examined Components,
No of Statements,
No of Comments,
No of Pre-processor Statements,
Complexity,
Functionality)

C5 method has been used first. Twelve rules have been generated: three for "effort for elimination of a defect less than one hour", eight for "effort for elimination of a defect more than one hour but less than one day" and one for "effort for elimination of a defect more than one day". A model generated by C5 gives 82.6% correct classifications for the training data. Once again two the most important rules generated for each output category are presented below.

Rule EE_C5_c1_1: **IF** defect type is *Clerical_Error*
THEN effort for elimination of a defect is less than 1 hour

Confidence level (bbm_T):	0.938
Data points satisfying the antecedent:	14
Data points satisfying the antecedent and with matching consequent:	14

Rule EE_C5_c1_2: **IF** defect type is *Requirements_Incorrect*

THEN effort for elimination of a defect is less than 1 hour

Confidence level (bbm_T): 0.667

Data points satisfying the antecedent: 1

Data points satisfying the antecedent and with matching consequent: 1

Rule EE_C5_c2_1: **IF** defect type is *Single_Design_Error*

AND lines of code is more than 135

THEN effort for elimination of a defect is more than 1 hour but less than 1 day

Confidence level (bbm_T): 0.875

Data points satisfying the antecedent: 6

Data points satisfying the antecedent and with matching consequent: 6

Rule EE_C5_c2_2: **IF** defect type is *Multiple_Design_Error*

THEN effort for elimination of a defect is more than 1 hour but less than 1 day

Confidence level (bbm_T): 0.833

Data points satisfying the antecedent: 4

Data points satisfying the antecedent and with matching consequent: 4

Rule EE_C5_c3_1: **IF** *Multiple_Components* examined

AND complexity is *Hard*

THEN effort for elimination of a defect is more than 1 day

Confidence level (bbm_T): 0.556

Data points satisfying the antecedent: 7

Data points satisfying the antecedent and with matching consequent: 4

4C, on the other hand, has generated seven rules for "effort for elimination of a defect less than one hour", nine for "effort for elimination of a defect more than one hour but less than one day",

and one for "effort for elimination of a defect more than one day". Classification rate is 89.54% for the training data. As above, only two rules for each case are presented:

Rule EE_4C_c1_1: **IF** defect type is *Single_Design_Error* **OR** *Language_Use_Error*
OR *Clerical_Error* **OR** *Requirements_Incorrect* **OR** *Functional_Spec._Incorrect* **OR** *Multiple_Error*
AND phase when error entered system is *Functional_Specification*
OR *Code_Testing*
AND *Single_Component* examined
AND number of comments in less than 63 **OR** more than 111
AND functionality is *Computational* **OR** *Control*
THEN effort for elimination of a defect is less than 1 hour

Confidence level (bbm_T): 0.900
Data points satisfying the antecedent: 18
Data points satisfying the antecedent and with matching consequent: 17

Rule EE_4C_c1_2: **IF** defect type is *Clerical_Error* **OR** *Requirements_Incorrect*
OR Multiple_Error
AND phase when error entered system is *Code_Testing*
AND complexity is *Easy* **OR** *Moderate*
THEN effort for elimination of a defect is less than 1 hour

Confidence level (bbm_T): 0.944
Data points satisfying the antecedent: 16
Data points satisfying the antecedent and with matching consequent: 16

Rule EE_4C_c2_1: **IF** defect type is *Multiple_Design_Error* **OR** *Language_Use_Error*
OR *Functional_Spec._Incorrect* **OR** *Other_Error*
AND number of comments in less than 130 **OR** more than 181
AND number of preprocessor statements is less than 5
OR more than 10
AND complexity is *Easy* **OR** *Moderate*
AND functionality is *Computational* **OR** *Data_Accessing*
OR *Error_Handling*
THEN effort for elimination of a defect is more than 1 hour

but less than 1 day

Confidence level (bbm_T): 0.882

Data points satisfying the antecedent: 15

Data points satisfying the antecedent and with matching consequent: 14

Rule EE_4C_c2_9: **IF** defect type is *Multiple_Design_Error* **OR** *Multiple_Error*
OR Other_Error
AND lines of code is less than 80
THEN effort for elimination of a defect is more than 1 hour
but less than 1 day

Confidence level (bba_T): 0.900

Data points satisfying the antecedent: 8

Data points satisfying the antecedent and with matching consequent: 8

Rule EE_4C_c3_1: **IF** defect type is *Language_Use_Error*
AND *Single_Component* examined
AND number of preprocessor statements is in the range <3, 5>
THEN effort for elimination of a defect is more than 1 day

Confidence level (bbm_T): 0.500

Data points satisfying the antecedent: 12

Data points satisfying the antecedent and with matching consequent: 6

Genetic-based approach has lead to nineteen rules: seven for "effort for elimination of a defect less than one hour" and twelve for "effort for elimination of a defect more than one hour but less than one day". Genetic approach has not generated any rules for "effort for elimination of a defect more than one day". Classification rate for the training is 79.1%. The rules are:

Rule EE_GB_c1_3: **IF** defect type is *Clerical_Error*
AND functionality is *Computational* **OR** *Data_Accessing*
THEN effort for elimination of a defect is less than 1 hour

Confidence level (bbm_T): 0.889

Data points satisfying the antecedent: 7

Data points satisfying the antecedent and with matching consequent: 7

Rule EE_GB_c1_7: **IF** defect type is *Clerical_Error*
AND functionality is *Control*
THEN effort for elimination of a defect is less than 1 hour

Confidence level (bbm_T): 0.889
Data points satisfying the antecedent: 7
Data points satisfying the antecedent and with matching consequent: 7

Rule EE_GB_c2_4: **IF** defect type is *Multiple_Error*
AND number of comments is less than 86
AND functionality is *Computational* **OR** *Data_Accessing*
THEN effort for elimination of a defect is more than 1 hour but less than 1 day

Confidence level (bbm_T): 0.833
Data points satisfying the antecedent: 4
Data points satisfying the antecedent and with matching consequent: 4

Rule EE_GB_c2_6: **IF** defect type is *Single_Design_Error* **OR** *Other_Error*
AND functionality is *Computational* **OR** *Data_Accessing*
THEN effort for elimination of a defect is more than 1 hour but less than 1 day

Confidence level (bbm_T): 0.875
Data points satisfying the antecedent: 6
Data points satisfying the antecedent and with matching consequent: 6

9.2 Validation of Rules

Validation of rules generated by C5 has brought only one change, Table 36. A rule EE_C5_c2_[illegible] has been replaced by a rule EE_C5_c2_4, which new updated value of bbm increased and suppress the updated value of bbm of the rule EE_C5_c2_2. A new rule is shown.

Table 36.Original and updated values of bbm for rules generated by C5

Rule Number	Original	After Validation
Category: less than 1 hour		
EE_C5_ c1_1	**0.938**	**0.974**
EE_C5_ c1_2	**0.667**	**0.800**
EE_C5_ c1_3	0.509	0.509
Category: more than 1 hour but less than 1 day		
EE_C5_ c2_1	**0.875**	**0.875**
EE_C5_ c2_2	**0.833**	0.833
EE_C5_ c2_3	0.800	0.800
EE_C5_ c2_4	0.727	**0.842**
EE_C5_ c2_5	0.714	0.625
EE_C5_ c2_6	0.700	0.609
EE_C5_ c2_7	0.625	0.625
EE_C5_ c2_8	0.571	0.727
Category: more than 1 day		
EE_C5_ c3_1	**0.556**	**0.714**

Rule EE_C5_c2_4: **IF** defect type is *Language_Use_Error*

AND complexity is *Easy*

THEN effort for elimination of a defect is more than 1 hour but less than 1 day

Confidence level (bbm_{UPDATE}): 0.842

Similar situation has occurred in the case of rules generated by 4C. Only one rule has been replaced by a new rule, Table 37. In this case, however, it has happened in the category "effort for elimination of a defect less than one hour". The rule EE_4C_c1_1 has been replaced by EE_4C_c1_6, which is shown below.

Table 37.Original and updated values of bbm for rules generated by 4C

Rule Number	Original	After Validation
Category: less than 1 hour		
EE_4C_c1_1	**0.900**	0.844
EE_4C_c1_2	**0.944**	**0.966**

EE_4C_c1_3	0.625	0.833
EE_4C_c1_4	0.538	0.538
EE_4C_c1_5	0.571	0.571
EE_4C_c1_6	0.857	**0.968**
EE_4C_c1_7	0.833	0.909
Category: more than 1 hour but less than 1 day		
EE_4C_c2_1	**0.882**	**0.938**
EE_4C_c2_2	0.550	0.550
EE_4C_c2_3	0.833	0.625
EE_4C_c2_4	0.833	0.833
EE_4C_c2_5	0.800	0.667
EE_4C_c2_6	0.750	0.750
EE_4C_c2_7	0.900	0.818
EE_4C_c2_8	0.500	0.333
EE_4C_c2_9	**0.900**	**0.947**
Category: more than 1 day		
EE_4C_c3_1	**0.500**	**0.500**

Rule EE_4C_c1_6: **IF** defect type is *Single_Design_Error* **OR** *Clerical_Error*
OR *Requirements_Incorrect*
AND lines of code is less than 130
AND functionality is *Computational* **OR** *Data_Accessing*
THEN effort for elimination of a defect is less than 1 hour

Confidence level (bbm_{UPDATE}): 0.968

Validation of the original set of rules generated by genetic-based method has also brought a single change. In the case of category "effort for elimination of a defect more than one hour but less than one day", an updated bbm value of the rule EE_GB_c2_4 has decreased. At the same time a new bbm value of the rule EE_GB_c2_1 has increased from 0.666 to 0.833 what has pushed this rule to the set of the best rules. The rule EE_GB_c2_1 is included.

Table 38.Original and updated values of bbm for rules generated by GB

Rule Number	Original	After Validation
Category: less than 1 hour		
EE_GB_c1_1	0.583	0.583
EE_GB_c1_2	0.667	0.800
EE_GB_c1_3	**0.889**	**0.970**
EE_GB_c1_4	0.667	0.667
EE_GB_c1_5	0.800	0.889
EE_GB_c1_6	0.714	0.556
EE_GB_c1_7	**0.889**	**0.889**
Category: more than 1 hour but less than 1 day		
EE_GB_c2_1	0.667	**0.833**
EE_GB_c2_2	0.647	0.647
EE_GB_c2_3	0.667	0.667
EE_GB_c2_4	**0.833**	0.714
EE_GB_c2_5	0.600	0.600
EE_GB_c2_6	**0.875**	**0.875**
EE_GB_c2_7	0.800	0.800
EE_GB_c2_8	0.750	0.750
EE_GB_c2_9	0.667	0.667
EE_GB_c2_10	0.750	0.750
EE_GB_c2_11	0.667	0.667
EE_GB_c2_12	0.667	0.667

Rule EE_GB_c2_1: **IF** defect type is *Language_Use_Error* **OR** *Functional_Spec._Incorrect*
AND number of comments is less than 86
AND functionality is *Computational* **OR** *Data_Accessing*
THEN effort for elimination of a defect is more than 1 hour but less than 1 day

Confidence level (bbm_{UPDATE}): 0.833

9.3 Analysis of IF-THEN Rules

As in the case of the previous experiment, let's examine the best **IF-THEN** rules (marked by bold fonts in (Table 36, Table 37 and Table 38) in "After Validation" column) which have been extracted to find relationships between software attributes, type of defect and when it has entered the system, and time needed for correction of a single defect.

For the category "effort for elimination of a defect less than one hour" the range of bbm values is from 0.800 to 0.974. Examination of the rules from this category points to three rules EE_C5_c1_1, EE_GB_c1_3 and EE_GB_c1_7. High values of bbm and similarly of the rules mean that they can be sum up with a very simple statement. If 'defect type' is *Clerical_Error* then time needed for correction of this defect is less than one hour.

Rule EE_C5_c1_1: **IF** defect type is *Clerical_Error*
THEN effort for elimination of a defect is less than 1 hour

Rule EE_GB_c1_3: **IF** defect type is *Clerical_Error*
AND functionality is *Computational* **OR** *Data_Accessing*
THEN effort for elimination of a defect is less than 1 hour

Rule EE_GB_c1_7: **IF** defect type is *Clerical_Error*
AND functionality is *Control*
THEN effort for elimination of a defect is less than 1 hour

In the category "effort for elimination of a defect more than one hour but less than one day" 4C-generated rules have the highest values of bbm. A rule EE_4C_c2_1 seems to be a very important one. Two other rules, C5-generated EE_C5_c2_4 and GB-generated EE_GB_c2_6, are very similar. The rule EE_C5_c2_4 overlaps with EE_4C_c2_1 in the case of attributes 'defect type' and 'complexity, when the rule EE_GB_c2_6 overlaps in the case of attributes 'defect type' and 'functionality'. All this and high bbm values indicate that these three rules can be useful in prediction of time needed for elimination of defects.

Rule EE_4C_c2_1: **IF** defect type is *Multiple_Design_Error* **OR** *Language_Use_Error*

OR *Functional_Spec._Incorrect* **OR** *Other_Error*
AND number of comments in less than 130 **OR** more than 181
AND number of preprocessor statements is less than 5
OR more than 10
AND complexity is *Easy* **OR** *Moderate*
AND functionality is *Computational* **OR** *Data_Accessing*
OR *Error_Handling*
THEN effort for elimination of a defect is more than 1 hour
but less than 1 day

Rule EE_C5_c2_4: **IF** defect type is *Language_Use_Error*
AND complexity is *Easy*
THEN effort for elimination of a defect is more than 1 hour
but less than 1 day

Rule EE_GB_c2_6: **IF** defect type is *Single_Design_Error* **OR** *Other_Error*
AND functionality is *Computational* **OR** *Data_Accessing*
THEN effort for elimination of a defect is more than 1 hour
but less than 1 day

The category "effort for elimination of a defect more than one day" is very poorly represented in the generated sets of rules. The rule generated by C5 is the best. Its bbm value is 0.714.

Rule EE_C5_c3_1: **IF** *Multiple_Components* examined
AND complexity is *Hard*
THEN effort for elimination of a defect is more than 1 day

It seems that one can say with a quite high confidence that if there is a need to examine a number of components to fully understand the impact of a defect, and complexity of software component where this defect has been found is *Hard* then elimination of such defect takes more than one day.

9.4 Prediction of Efforts for Elimination of Defects

Prediction of efforts for elimination of defects for the testing data points has been performed using the multi-model estimation system. Classification rate and confidence levels assigned to obtained classifications have been recorded.

The testing set contains 23 data points. Using only a single set of rules generated by C5, 4C and GB the classification rates are 56.52%, 56.52% and 65.22% respectively (as it has been mentioned before some data points have not satisfied any rules). Application of the multi-model estimation system increased classification rate to 78.26% (it is translated to eighteen proper classifications, comparing to fifteen obtained for GB rules). In the case of effort prediction, twelve data points satisfy rules that indicate a contradicting classification. Eleven remaining data points are unanimously classified by all rules that are part of the system. Confidence levels of these classifications are around 0.99. The rules that are satisfied for these twelve points are presented in Table 39.

Table 39.Rules satisfied by the twelve "confusing" data points

	C5		4C		GB		Original Category	Predicted Category
	I	II	I	II	I	II		
1	c1_4	c2_4	c1_3		c1_10		I	I
2	c1_4		c1_5	c2_2	c1_4		III	II
3	c1_4					c2_3	I	I
4	c1_4			c2_1	c1_13		II	II
5	c1_1			c2_1	c1_12		II	II
6		c2_7	c1_2	c2_1		c2_7	III	III
7	c1_4	c2_5		c2_3		c2_3	I	II
8	c1_4			c2_1	c1_13		II	II
9	c1_1			c2_1	c1_12		II	II
10		c2_7	c1_2	c2_1		c2_7	I	I
11	c1_4	c2_5		c2_3		c2_3	II	I
12	c1_4	c2_5		c2_3		c2_3	I	I

It can be easily observed that each of these points is not univocally classified to a single category. Rules generated by C5 have problems with data points number 1, 2, 3, 4, 7, 9 and 11, rules generated by 4C are misled for points number 1, 2, 4, 5, 6, 7, 8, 10, 11 and 12, in the case of rules genetically generated points number 2, 3, 6, 7 and 11 are wrongly classified. Application of TBM has resulted in BetP values. Table 40 and Table 41 show these values for each category for each of the twelve data points.

Table 40.BetP values for the first six data points

	Data Points					
	1	**2**	**3**	**4**	**5**	**6**
BetP (Category I)	0.638	0.031	0.599	0.055	0.001	0.108
BetP (Category II)	0.308	0.962	0.381	0.932	0.996	0.434
BetP (Category III)	0.054	0.007	0.020	0.013	0.003	0.458

Table 41.BetP values for the last six data points

	Data Points					
	7	**8**	**9**	**10**	**11**	**12**
BetP (Category I)	0.119	0.014	0.002	0.999	0.673	0.999
BetP (Category II)	0.878	0.931	0.998	0.001	0.309	0.001
BetP (Category III)	0.003	0.055	0.000	0.000	0.018	0.000

A special attention should be paid to data points number 3 and 6. In the case of a point number 3, rules generated by C5 and GB indicate membership of this point to category I, however a higher bbm value of a 4C rule (comparing to bba's of C5 and GB rules) has changed a verdict of the System. It can be seen that probability of the point belonging to category II is relatively high 0.878. The point number 6, on the other hand, satisfies one category I rule, two category II rules and two category III rules. Application of the System in such a case classified the data point to a proper category. However, in the case that probabilities are concern, it can be seen that BetP(category III) is only slightly larger than BetP(category II).

In the case of point number 7, a high bbm value of GB rule EE_GB_c2_1 (supported by bbm values of EE_C5_c2_6 and EE_4C_c2_4) has caused misclassification; rules EE_C5_c1_3 and EE_4C_c1_1 have too small bbm values to increase the value of BetP for Category I.

10 Conclusions and future work

Knowledge extraction has shown to be an important technique of analysis of software engineering data. This is a significant step in the continued drive towards gaining as much knowledge as possible from data. The constant increase in software and hardware infrastructure will only increase the availability of data in software organizations. There has been an increase in the number and quality of tools available for data analysis due to an increase in knowledge extraction research. The use of knowledge extraction techniques is on the increase among software engineering practitioners and data analysts.

The availability of new data analysis methods and tools should be met with caution by software engineering professionals. First of all, these techniques can only be as good as the data collected. Having good data is the first requirement for good data exploration. There can be no good knowledge discovery on bad data. This thesis does not discuss data collection improvement, but the issue should be on the top of the list for any organization planning to start a data-mining program.

In this study, a new concept has been proposed to analyze software engineering data. It applies elements of evidence theory and a combination of rule-based models in order to create an estimation system, which has the capability to determine confidence levels of the generated predictions. The application of a number of rule-based models developed with different techniques enhances prediction capabilities and allows more comprehensive extraction of knowledge from data. Confidence levels associated with predictions and extracted knowledge allows users to be "more critical" about extracted knowledge. It leads to a better understanding of the relationships between attributes of data and the mechanisms, which generate the data, i.e. software development processes.

The proposed system has been applied to analysis of software corrective maintenance data. The purpose of this analysis has been to explore and understand the relationship between attributes of software, types of defects and efforts needed for conducting defect elimination tasks. In this work, special attention has been given to the number of software components that have to be examined

to remove a single defect, the total effort needed for removal of the defect and the total time needed for isolation of defects.

The study has indicated some important rules in these three areas with high confidence values, which can be useful for prediction proposes. For the data set, 'time needed for isolation of defects', the rule: if 'defect type' is *Clerical_Error* or *Functional_Specification_Incorrect* then 'time to isolate defect' is *less than one hour.* This rule has a confidence level of 0.972. Also if 'defect type' is *Language_Use_Error* and 'number of comments' is *more than 82* or *less than 132* and 'complexity' is *Easy* then, 'time to isolate defect' is *more than 1 hour* with a confidence level of 0.957.

For the data set, 'effort for elimination of defects', the rule: if 'defect type' is *Clerical_Error,* then 'effort for elimination of defect' is *less than 1 hour* with confidence level of 0.974 and if *Multiple_Components* are examined and 'complexity' is *Hard,* then 'effort for elimination of defect' is *more than 1 day* with a confidence level of 0.714.

For the data set 'number of components examined', if 'defect type' is *Multiple_Design_Error* or *Clerical_Error* or *Functional_Spec._Incorrect* and 'functionality' is *Error_Handling* or *Control* or *Data_Accessing* then, the result is 'single component examined' with a confidence level of 0.988. If 'defect type' is *Functional_Spec._Incorrect* and 'lines of code' is *more than 134* and 'complexity' is *Easy* then the result is 'multiple components examined' with a confidence level of 0.889.

These rules provide valuable information about relationships between different software attributes and resources required by maintenance activities. Usage of such information by maintenance engineers can lead to more accurate planning of maintenance tasks.

Application of the multi-model estimation system increased the classification rate by about '10 - 23%' for the data set of 'time to isolate defects', '13 - 17%' for the data set of 'number of component examined' and '13 - 22%' for the data set of 'efforts for elimination of defects'. It should also be noted that this approach has also helped in resolving conflicts in classification from these three models (4C, C5 and GB). For instance, in cases where the prediction on a particular data point is different for each model, the multi-model system helps indicate which group it actually belongs to with a given probability. Estimation systems, extracted knowledge and results of prediction processes have been described and investigated.

This research was faced with several limitations, which may have affected the performance of the model developed and confidence in the obtained results. One of the limitations was that some of the data used for this study were missing so those data points were removed from the data set. This can be due to the collection process. It is hoped that getting a complete data would improve knowledge extraction process and lead to discovering true relationships between data. Also, the data used for this thesis was pretty small, this might be a limitation. It would be interesting to do the same analysis with large software maintenance data.

It is believed that other methods of using multi-models like bagging and boosting produces a reasonable improvement over single models. It would be interesting to combine bagging and boosting to the approach proposed in this thesis to see the improvement that will be gained by using the resulting multi-multi-model.

Finally, A pressing question for future work in the area of multiple models research is the following:

> *Are multiple models able to do better than a single model because they involve more searches or is it because multiple models are retained for classification?*

To test this empirically, it would be necessary to construct a learning algorithm that does as much search as the multiple models method but only produces a single model. That method would only differ from the multiple models method in one variable: the number of models retained for classification. That achieved, it would then be possible to compare the performance of the two methods on many data sets and if on some data sets the multiple models method performed significantly better, one could definitively conclude that the multiple models method does not do better simply because it performs more searches but that its ability to retain multiple models for classification is crucial (at least for that group of data sets).

References

1. 4cData,LLC Main Page http://www.4cdata.net (March 2003)

2. Ali K. A comparison of methods for learning and combining evidence from multiple models

3. Ali K. and Pazzani, M. ``Error Reduction through Learning Multiple Descriptions", in press, Machine Learning.

4. Arthur, L. J, Software Evolution. New York: John Wiley & Sons, Inc., 1998.

5. Breiman, L. Bagging Predictors. Machine Learning 24(2): 123-140, 1996.

6. Briand, C. L., "On the many ways Software Engineering can benefit from Knowledge Engineering, Invited Keynote Address, Proceedings of ACM SEKE conference, 2002.

7. Buntine, Wray 1990. A Theory of Learning Classification Rules. Ph.D. Dissertation, University of Technology, Sydney.

8. Chatzoglou, P.D. and Macaulay L.A.. A Rule-Based Approach to Developing Software Development Prediction Models. Automated Software Engineering 5, 1998, pages 211-243.

9. Evanco, W. M. Prediction Models for Software Fault Correction Effort. Proceedings of the Fifth Conference on Software Maintenance and Reengineering, CSMR 2001, 14-16 March 2001, Lisbon, Portugal, pages114-120

10. Fenton, N.E., Neil, M.. A Critique of Software Defect Prediction Models, IEEE Transactions of Software Engineering, 25(3), 1999, pages

11. Freund, Y and Schapire, R. E. (1996) Experiments with a New Boosting Algorithm. In proceedings of the Thirteenth International Conference on Machine Learning. Morgan Kaufmann.

12. Glass, R. Software maintenance documentation. Annual ACM Conference on Systems Documentation, Pittsburgh, PA, 1989.

13. Jorgensen, Magne., "Experience With the Accuracy of Software Maintenance Task Effort Prediction Models", IEEE Transactions on Software Engineering, June 1995, p. 674-681.

14. Kononenko and M. Kovacic M, ``Learning as Optimization: Stochastic Generation of Multiple Knowledge," Proc. 9th International Workshop on Machine Learning, Aberdeen, UK: Morgan Kaufmann, pp. 257-262, 1992.

15. Kwok S. and Carter C., Multiple decision trees," Uncertainty in Artificial Intelligence, Vol. 4, pp. 327-335, 1990.

16. Mendonca M., Sunderhaft N. L. Mining Software Engineering Data: A Survey: A DACS State-of-the-Art Report, 1999.

17. Osek, J. T., and Palvia, P. "Software Maintenance management: Changes in the Last Decade," Journal of Software Maintenance, 2(3), September 1990, p. 157-174.

18. Power, D. J., What is a DSS?. The Online Executive Journal for Data-Intensive Decision Support, October 21, 1997; Vol. 1, No. 3.

19. Quinlan J.R. C4.5: Programs for machine learning. Morgan Kaufmann Publishers, 1993.

20. Reformat M., Pedrycz W., and Pizzi, N. Software Quality Analysis with the use of Computational Intelligence (Extended version), to appear in Information and Software Technology, a special issue on Computational Intelligence in Software Engineering.

21. RuleQuest Research Data Mining Tools http://www.rulequest.com (March, 2003).

22. Savage, L.J.. Foundations of Statistics. Wiley New York, 1954.

23. Schneidewind, N. F., "The State of Software Maintenance", IEEE Transactions on Software Engineering, SE-13 (3), March 1987, p. 303-310.

24. Shafer, G. A mathematical Theory of Evidence. Princeton University Press, 1976.

25. Shafer, G. Theory of Evidence Home Page http://www.dei.unipd.it/~cuzzolin/Belief.htm (March, 2003).

26. Slaughter, A. S., and R. D. Banker, " A Study of the Effects of Software Development Practices on Software Maintenance Effort" IEEE Journal of Software Maintenance, 1996, p.197-205.

27. Smets, Ph. and Kennes, R. The Transferable Belief Model. Artificial Intelligence, 66, 1994, pages191-234.

28. Smets, Ph. Belief Functions. In Non Standard Logics for Automated Reasoning, Academic Press, London 1988, pages 253-286.

29. Smets, Ph. The Application of the Transferable Belief Model to Diagnostic Problems, July 1999, pp 1-12.

30. Smets, Ph.. The concept of distinct evidence, IPMU 92 Proceedings, 1992, pages 253-286.

31. Smith, D.. Designing maintainable software. New York, Springer, 1999.

32. Todorovski L, Dzeroski S. Combining Multiple Models with Meta Decision Trees. In *Proceedings of the Fourth European Conference on Principles of Data Mining and Knowledge Discovery.* Springer, 2000; 54-64.

33. UCI Machine Learning Repository http://www.ics.uci.edu/~mlearn/MLRepository.html (March, 2003)

34. Winer, B. J. Statistical Principles in Experimental Design, 1992, Page 9-10

35. Zvegintzov, N., "Immortal Software," Datamation, June 1984, p170-180.